DEVICE AND METHOD FOR ADDING AND DETECTING ELECTRONIC INFORMATION

Publication number: JP11259067

Publication date: 1999-09-24

Inventor: TERADA YOSHINARI; TARUDA HIDEAKI

Applicant: YAMAHA CORP

Classification:

- international: ***G10H1/00; G09C5/00; G10K15/02; H03M7/00;*** **G10H1/00; G09C5/00; G10K15/02; H03M7/00;** (IPC1-7): G10H1/00; H03M7/00

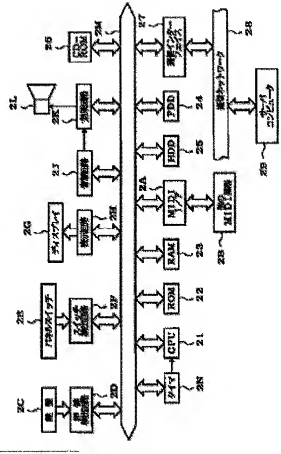
- European:

Application number: JP19980346150 19981204

Priority number(s): JP19980346150 19981204; JP19970350126 19971204

Abstract of **JP11259067**

PROBLEM TO BE SOLVED: To add attached information without altering a data format and to process these data with code by operating predetermined calculations to a part of main information by using attached information and externally settable auxiliary information. SOLUTION: An electronic instrument operating as an electronic information processing system stores an electronic signature, namely, a copyright display data, which is a part of header information of MIDI data read out of a floppy disk drive 24, by dispersing it in a key-on event data in a relatively large data unit, and enciphers the data with the MIDI format maintained. Namely, scramble decoding data, showing which encipherment processing of plural ones is operated thereto, are embedded in the head of the MIDI data, which is not subjected to encipherment processing but is arranged so that ordinary MIDI data can be reproduced, and the remaining MIDI data part are subjected to scramble processing corresponding to scramble decoding data to disable the reproduction of the data.



(19)日本国特許庁（ＪＰ）　(12)公開特許公報（A）　(11)特許出願公開番号

特開平11－259067

(43)公開日　平成11年(1999)９月24日

(51)Int.Cl.⁶	識別記号	ＦＩ		
Ｇ１０Ｈ　1/00		Ｃ１０Ｈ	1/00	Ｚ
Ｈ０３Ｍ　7/00		Ｈ０３Ｍ	7/00	

審査請求　未請求　請求項の数４　ＯＬ　（全 36 頁）

(21)出願番号　特願平10－346150

(22)出願日　平成10年(1998)12月４日

(31)優先権主張番号　特願平9－350126

(32)優先日　平９(1997)12月４日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000004075
ヤマハ株式会社
静岡県浜松市中沢町10番１号

(72)発明者　寺田　好成
静岡県浜松市中沢町10番１号　ヤマハ株式会社内

(72)発明者　樽口　秀昭
静岡県浜松市中沢町10番１号　ヤマハ株式会社内

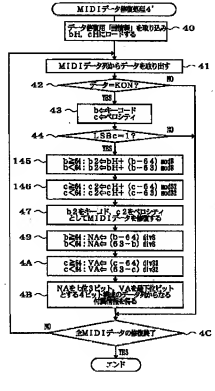
(74)代理人　弁理士　飯塚　義仁

(54)【発明の名称】　電子情報付与及び検出のための装置及び方法

(57)【要約】

【課題】　ＭＩＤＩデータ等の主要情報内に電子署名等の付属情報を電子透しにより埋め込むこと。その解読のための鍵情報のセキュリティ確保。

【解決手段】　電子情報付与装置（エンコーダ）の側において、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に溶ませてなるデータファイルを提供する際に、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部が、該データファイルの中に含まれないようにする。また、電子情報検出装置（デコーダ）の側において、前記鍵情報の少なくとも一部を、外部から供給させる情報として受け取る鍵情報受取手段を具備し、受け取った鍵情報を利用して付属情報の抽出若しくは主要情報の修復を行なう。



【特許請求の範囲】

【請求項１】　主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませてなるデータファイルを提供する電子情報付与装置において、

前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なう手段を具備する電子情報付与装置。

【請求項２】　主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませてなるデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復する電子情報検出装置において、

前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取る鍵情報受取手段を具備し、受け取った鍵情報を利用して前記付属情報の抽出若しくは前記主要情報の修復を行なうようにしたことを特徴とする電子情報検出装置。

【請求項３】　主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませてなるデータファイルを提供する電子情報付与方法において、

前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なうステップを具備する電子情報付与方法。

【請求項４】　主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませてなるデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復する電子情報検出方法において、

前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取るステップを具備し、受け取った鍵情報を利用して前記付属情報の抽出若しくは前記主要情報の修復を行なうようにしたことを特徴とする電子情報検出方法。

【発明の詳細な説明】

【０００１】

【発明の属する技術分野】この発明は電子的記憶媒体に楽音制御情報（ＭＩＤＩデータなど）などの主要情報と付属情報とを暗号化処理して記憶したり、記憶媒体に記憶されている暗号化処理された主要情報と付属情報に基づいて元の主要情報及び付属情報を検出する電子情報付与及び検出のための装置及び方法に関する。

【０００２】

【従来の技術】最近では、パーソナルコンピュータを使用して、ユーザ自身が音楽データ、映像データ及び波形データなどを作成したり、それらに種々の変更を加えたりすることが容易にできるようになった。従って、パーソナルコンピュータを使用することによって、市販のＦＤ、ＣＤ－ＲＯＭ、ＬＤなどの記憶媒体に記録された音楽データ、映像データ及び波形データなどを自由に読み出して、それらに種々の変更処理を加えたりすることができる。市販のＣＤ－ＲＯＭやＬＤなどに記録されているデータは、その販売者や製作者などに著作権があるため、本来それらを自由に改変することは、著作権侵害の観点から許されるものではない。従って、現在では、ＣＤ－ＲＯＭやＬＤなどの媒体に記録されている音楽データ、映像データ又は波形データなど主データの著作物がだれに属するものなのかを示すための著作権表示データを、その音楽データ、映像データ又は波形データなど主データを記録する主データ記録部とは別のヘッダ部において付属情報として付加的に記録することによって、著作権者を明示し、著作権侵害の未然防止を図っているのが現状である。また、付属情報には、このような著作権表示データの他にも、その音楽データ、映像データ及び波形データの題名などを示す情報、又はその映像データや波形データなどがどのようなデータ圧縮技術で圧縮されているのかなどの記録形式を示す情報などがある。

【０００３】

【発明が解決しようとする課題】しかしながら、現在では、パーソナルコンピュータなどを使って自由にデータを書き換えたり変更したりすることができるので、故意又は過失によって、著作権表示データや種々の付属情報が削除されたり、書き換えられたりするという問題がある。特に、これらの付属情報がヘッダ部に記録されている場合は、削除や改竄が容易になされてしまうという問題がある。また、最近ではネットワーク通信の発達に伴って、このように著作権表示データの削除や改竄がなされた音楽データ、映像データ又は波形データなどが通信ネットワークを介して広く流通するという問題も起こり易い。この発明は、音楽データ、映像データ又は波形データなどのような主要情報のデータフォーマットを変更することなく付属情報を付加すると共にこれらのデータに暗号化処理を施し、この暗号を解読しない限り、こられの主要情報や付属情報を再生して利用することのできないようにした電子情報付与及び検出装置及び方法に関するものである。

【０００４】

【課題を解決するための手段】この発明に係る電子情報付与装置は、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませてなるデータファイルを提供する電子情報付与装置において、前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なう手段を具備するものである。前記補助情報は外部から任意に設定可能であり、設定された該補助情報は、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な「鍵情報」となりうるものである。これにより、解読の際の鍵情報となる前記補助情報そのものは、データファイルのデータ内容とは無関係に任意に設定されるから、該データファイルの中には直接的にはそれと判るように含まれていないものとなる。なお、付属情報を主要情報の中に潜在的に組み込むためのアルゴリズム演算に際して使用する補助情報としては、データを暗号化するための演算に使用する変数情報や、データを暗号化するための演算を行う際の種々の条件を設定する変数情報若しくは条件設定情報など、その他、付属情報を主要情報の中に潜在的に組み込むためのアルゴリズム演算に関連して使用される任意の情報であってよい。

【０００５】この電子情報付与装置は、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませてなるデータファイルを提供するものであり、換言すれば、主要情報の中に付属情報を潜在的に埋め込み、主要情報に対して結果的に暗号化処理を施すことになるものである。一例として、主要情報としては、ＭＩＤＩデータのキーオンイベントデータ、プログラムチェンジデータ又はコントロールチェンジデータ、波形データ、又は映像データなどがある。付属情報としては、著作者名、曲の題名、画像／映像の題名などに関する文字データや波形データの圧縮方法などのデータ形式に関するデータやその他の種々のデータ（暗号文、鍵情報、ＩＤ、パスワード、ニュース文）などがある。この発明に係る電子情報付与装置（換言すれば「エンコーダ」）と電子情報検出装置（換言すれば「デコーダ」）とを総称して電子情報処理システムと呼ぶとすると、この電子情報処理システムでは、一例として、主要情報を構成するデータ群の中の所定のデータ群として、ＭＩＤＩデータの場合にはキーオンイベントデータ群を用い、このキーオンイベントデータ群の中の単位データとして、ベロシティデータやキーコードデータを用い、これらのデータのうち所定のデータを付属情報のデータ部分に応じて分散的に変更し、主要情報のデータ群の中に付属情報のデータを分散的に潜ませる（埋め込む）ことで、付属情報を主要情報のデータ群の中に潜ませてなるデータファイルを提供する。

【０００６】このデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復するためには、解読のために所定の「鍵」情報が必要となるが、そのための「鍵」情報それ自体が該データファイルの中に含まれているような記録形態もありうる。しかし、それでは、オーソライズされていない者が、不当に当該データファイルの再生を行ないまたその付属情報を解読しようとする場合に、当該データファイルの中から解読のために必要な所定の「鍵」情報を容易に取得してしまうおそれがあり、セキュリティ面での信頼性に欠けるものとなってしまう。そこで、この発明では、所定のアルゴリズムに従う演算を前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なうようにしたことにより、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な「鍵情報」の一種となる前記補助情報そのものは、データファイルのデータ内容とは無関係に任意に設定されることで、該データファイルの中には直接的にはそれと判るように含まれていないものとなるようにしたことを特徴としたものである。これにより、オーソライズされていない者が、不当に当該データファイルから主要情報の修復を行ないまたその付属情報を解読しようとしたとしても、肝心のデータ修復用「鍵情報」をデータファイルからは得ることができないことにより、データ修復若しくは解読を行なうことができないことになり、暗号化又は電子透し埋め込みによる機密的な情報記録に際して、セキュリティ機能を向上させることができる、という優れた効果を奏する。

【０００７】この発明に係る電子情報検出装置は、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更することで該付属情報を主要情報のデータ群の中に潜ませてなるデータファイルから前記付属情報を抽出する若しくは前記主要情報を修復する電子情報検出装置において、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取る鍵情報受取手段を具備し、受け取った鍵情報を利用して前記付属情報の抽出若しくは前記主要情報の修復を行なうようにしたことを特徴とするものである。

【０００８】このように、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復する機能を有する電子情報検出装置の側において、鍵情報受取手段を具備することにより、前記データファイルから前記付属情報を抽出する若しくは前記主要情報を修復するために必要な鍵情報の少なくとも一部を、外部から供給される情報として受け取り、この受け取った鍵情報を利

用して前記付属情報の抽出若しくは前記主要情報の修復を行なうことができるようになる。すなわち、上述と同様に、データファイルの中には必要な鍵情報が含まれていないことになり、オーソライズされていない者が、不当に当該データファイルから主要情報の修復を行ないまたその付属情報を解読しようとしたとしても、肝心のデータ修復用「鍵情報」をデータファイルからは得ることができないことにより、データ修復若しくは解読を行なうことができないことになり、暗号化又は電子透し埋め込みによる機密的な情報記録に際して、セキュリティ機能を向上させることができる、という優れた効果を奏する。鍵情報受取手段に対する鍵情報の与え方は、どのようなやり方を用いてもよい。例えば、データファイルとは別途にデータ通信等により電子情報の形で自動的に与えるようにしてもよいし、オーソライズされた利用者に対してのみ印刷物または電話等で通知することに基づき該利用者が該鍵情報のデータ入力操作を手動で行なうようにしてもよいし、様々なやり方であってよい。

【０００９】なお、付属情報を主要情報の中に潜在的に組み込むためのアルゴリズム演算に際して使用する「補助情報」、若しくは前記付属情報を抽出する又は前記主要情報を修復するのに必要とされる「鍵情報」は、一種類とは限らず、複数種類あってよい。例えば、電子情報付与装置（エンコーダ）において、前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するために、前記主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと所定の変数とを用いて行なうようにした場合、その解読（デコード）のためにアルゴリズムのみならず、該所定の変数も必要であり、この所定の変数に対応する情報が上記補助情報若しくは鍵情報の一種として使用されることとなる。あるいは、前記主要情報を構成するデータ群の中の一部のデータを前記付属情報を構成するデータに基づいて変更するためのデータ編集処理を行なうときに、特定の編集条件を設定して該データ編集処理を行なったような場合も、その解読（デコード）のためには設定された編集条件が何であったかが判っている必要があり、この編集条件を設定する情報も上記補助情報又は鍵情報の一種として使用されることとなる。

【００１０】この発明は、装置発明として構成し、実施することができるのみならず、方法発明として構成し、実施することもできる。また、この発明は、コンピュータプログラムまたはＤＳＰ等のためのマイクロプログラムの形態で実施することができるし、そのようなコンピュータプログラムまたはマイクロプログラムを記憶した記録媒体の形態で実施することもできる。

【００１１】

【発明の実施の形態】以下、この発明の実施の形態を添付図面に従って詳細に説明する。図２は、この発明に係る電子情報処理システムとして動作する電子楽器の全体構成を示すブロック図である。マイクロプロセッサユニット（ＣＰＵ）２１はこの電子楽器全体の動作を制御するものである。このＣＰＵ２１に対して、バス２Ｍを介して各種のデバイスが接続される。

【００１２】ＣＰＵ２１はＲＯＭ２２及びＲＡＭ２３内の各種プログラムや各種データ、及び外部記憶装置から取り込まれた楽音制御情報（ＭＩＤＩデータ）に基づいて全体の動作を制御すると共に取り込まれたＭＩＤＩデータ内に付属情報を分散して記憶したり、付属情報の分散記憶されたＭＩＤＩデータをフロッピーディスクから取り込み、それから付属情報を検出したりする。この実施の形態では、外部記憶装置として、フロッピーディスクドライブ２４、ハードディスクドライブ２５、ＣＤ－ＲＯＭドライブ２６などを例に説明するが、これ以外の光磁気ディスク（ＭＯ）ドライブ、ＰＤドライブなどを用いてもよい。また、通信インターフェイス２７を介して通信ネットワーク２８上のサーバコンピュータ２９などから楽音制御情報などの各種情報などを取り込んでもよいし、ＭＩＤＩインターフェイス２Ａを介して他のＭＩＤＩ機器２Ｂなどから ＭＩＤＩデータなどを取り込んでもよい。ＣＰＵ２１は、このような外部記憶装置から取り込まれたＭＩＤＩデータや鍵盤２Ｃの押鍵操作に基づいて生成したＭＩＤＩデータを音源回路２Ｊに供給する。なお、外部に接続された音源回路を用いて発音処理を行うようにしてもよい。

【００１３】ＲＯＭ２２はＣＰＵ２１の各種プログラム（システムプログラムや動作プログラムなど）や各種データを格納するものであり、リードオンリーメモリ（ＲＯＭ）で構成されている。ＲＡＭ２３は、ＣＰＵ２１がプログラムを実行する際に発生する各種データを一時的に記憶するものであり、ランダムアクセスメモリ（ＲＡＭ）の所定のアドレス領域がそれぞれ割り当てられ、レジスタ、バッファ、フラグ、テーブル等として利用される。

【００１４】また、ハードディスク装置２５などの外部記憶装置に前記動作プログラムを記憶するようにしてもよい。また、前記ＲＯＭ２２に動作プログラムを記憶せずに、ハードディスク装置２５などの外部記憶装置にこれらの動作プログラムを記憶しておき、それをＲＡＭ２３に読み込むことにより、ＲＯＭ２２に動作プログラムを記憶したときと同様の動作をＣＰＵ２１に行わせるようにしてもよい。このようにすると、動作プログラムの追加やバージョンアップ等が容易に行える。着脱自在な外部記憶媒体の１つとして、ＣＤ－ＲＯＭを使用してもよい。このＣＤ－ＲＯＭには、自動演奏データやコード進行データや楽音波形データや映像データなどの各種データ及び任意の動作プログラムを記憶していてもよい。ＣＤ－ＲＯＭに記憶されている動作プログラムや各種データは、ＣＤ－ＲＯＭドライブ２６によって、読み出さ

れ、ハードディスク装置２５に転送記憶させることができる。これにより、動作プログラムの新規のインストールやバージョンアップを容易に行うことができる。

【００１５】なお、通信インターフェイス２７をデータ及びアドレスバス２Ｍに接続し、この通信インターフェイス２７を介してＬＡＮ（ローカルエリアネットワーク）やインターネット、電話回線などの種々の通信ネットワーク２８上に接続可能とし、他のサーバコンピュータ２９との間でデータのやりとりを行うようにしてもよい。これにより、ハードディスク装置２５内に動作プログラムや各種データが記憶されていないような場合には、サーバコンピュータ２９からその動作プログラムや各種データをダウンロードすることができる。この場合、クライアントとなる楽音生成装置である自動演奏装置から、通信インターフェイス２７及び通信ネットワーク２８を介してサーバコンピュータ２９に動作プログラムや各種データのダウンロードを要求するコマンドを送信する。サーバコンピュータ２９は、このコマンドに応じて、所定の動作プログラムやデータを、通信ネットワーク２８を介して自動演奏装置に送信する。自動演奏装置では、通信インターフェイス２７を介してこれらの動作プログラムやデータを受信して、ハードディスク装置２５にこれらのプログラムやデータを蓄積する。これによって、動作プログラム及び各種データのダウンロードが完了する。

【００１６】なお、本発明は、本発明に対応する動作プログラムや各種データをインストールした市販のパーソナルコンピュータ等によって、実施させるようにしてもよい。その場合には、本発明に対応する動作プログラムや各種データを、ＣＤ－ＲＯＭ２６やフロッピーディスク等の、パーソナルコンピュータが読み込むことができる記憶媒体に記憶させた状態で、ユーザーに提供してもよい。そのパーソナルコンピュータが読み込むことができる記憶媒体に記憶させた状態で、ユーザに提供してもよい。そのパーソナルコンピュータ等が、ＬＡＮ、インターネット、電話回線等の通信ネットワークに接続されている場合には、通信ネットワークを介して、動作プログラムや各種データ等をパーソナルコンピュータ等に提供してもよい。

【００１７】鍵盤２Ｃは発音すべき楽音の音高を選択するための複数の鍵を備えており、各鍵に対応したキースイッチを有しており、また必要に応じて押圧力検出装置等のタッチ検出手段を有している。鍵盤２Ｃは音楽演奏のための基本的な操作子であり、これ以外の演奏操作子でもよいことはいうまでもない。押鍵検出回路２Ｄは発生すべき楽音の音高を指定する鍵盤２Ｃのそれぞれの鍵に対応して設けられたキースイッチ回路を含むものである。この押鍵検出回路２Ｄは鍵盤２Ｃの離鍵状態から押鍵状態への変化を検出してキーオンイベントを出力し、押鍵状態から離鍵状態への変化を検出してキーオフイベントを出力すると共にそれぞれのキーオンイベント及びキーオフイベントに関する鍵の音高を示すノートナンバを出力する。押鍵検出回路２Ｄはこの他にも鍵押し下げ時の押鍵操作速度や押圧力等を判別してベロシティデータやアフタタッチデータを出力する。

【００１８】パネルスイッチ２Ｅは自動演奏スタート／ストップスイッチ、一時停止（ポーズ）スイッチ、音色、音量、効果等を選択、設定、制御するための各種スイッチを含むものである。スイッチ検出回路２Ｆはパネルスイッチ２Ｅ上の各スイッチ群に対応して設けられており、これらの各スイッチ群の操作状態に応じたスイッチイベントをバス２Ｍを介してＣＰＵ２１に出力する。表示回路２ＨはＣＰＵ２１の制御状態、設定データの内容等の各種の情報をディスプレイ２Ｇに表示するものである。この実施の形態では、著作権表示データ、曲名、作曲者情報、作成年月日、歌詞、ニュース文、機種名（ハード名）、ＩＤなどのテキストデータなどを表示する。ディスプレイ２Ｇは液晶表示パネル（ＬＣＤ）等から構成され、表示回路２Ｈによってその表示動作を制御される。

【００１９】音源回路２Ｊは、複数のチャンネルで楽音信号の同時発生が可能であり、ＣＰＵ２１から与えられた楽音制御情報（ノートオン、ノートオフ、ベロシティ、ピッチデータ、音色番号等のＭＩＤＩデータ）を入力し、これらの楽音制御情報に基づいた楽音信号を発生する。音源回路２Ｊにおいて複数チャンネルで楽音信号を同時に発音させる構成としては、１つの回路を時分割で使用することによって複数の発音チャンネルを形成するようなものや、１つの発音チャンネルが１つの回路で構成されるような形式のものであってもよい。また、音源回路２Ｊにおける楽音信号発生方式はいかなるものを用いてもよい。例えば、発生すべき楽音の音高に対応して変化するアドレスデータに応じて波形メモリに記憶した楽音波形サンプル値データを順次読み出すメモリ読み出し方式（波形メモリ方式）、又は上記アドレスデータを位相角パラメータデータとして所定の周波数変調演算を実行して楽音波形サンプル値データを求めるＦＭ方式、あるいは上記アドレスデータを位相角パラメータデータとして所定の振幅変調演算を実行して楽音波形サンプル値データを求めるＡＭ方式等の公知の方式を適宜採用してもよい。また、これらの方式以外にも、自然楽器の発音原理を模したアルゴリズムにより楽音波形を合成する物理モデル方式、基本波に複数の高調波を加算することで楽音波形を合成する高調波合成方式、特定のスペクトル分布を有するフォルマント波形を用いて楽音波形を合成するフォルマント合成方式、ＶＣＯ、ＶＣＦ及びＶＣＡを用いたアナログシンセサイザ方式等を採用してもよい。また、専用のハードウェアを用いて音源回路を構成するものに限らず、ＤＳＰとマイクロプログラムを用いて音源回路を構成するようにしてもよいし、ＣＰＵ

とソフトウェアのプログラムで音源回路を構成するようにしてもよい。

【0020】タイマ2Nは時間間隔を計数したり、自動演奏のテンポを設定したりするためのテンポクロックパルスを発生する。このテンポクロックパルスの周波数は各種スイッチ2Eの中のテンポ設定スイッチや予め演奏データに含まれている図3のようなテンポデータに基づいて決定されている。タイマからのテンポクロックパルスはCPU21に対してインタラプト命令として与えられ、CPU21はインタラプト処理により自動演奏時における各種の処理を実行する。効果回路2Kは音源回路2Jからの楽音信号に種々の効果を付与し、効果の付与された楽音信号をサウンドシステム2Lに出力する。効果回路2Kによって効果の付与された楽音信号は、アンプ及びスピーカからなるサウンドシステム2Lを介して発音される。

【0021】次に、この発明に係る電子情報処理システムが電子情報を付与すると共に暗号化処理を同時に行う場合の動作の一例について説明する。図3は、電子楽器1が電子情報処理システムとして動作する場合のMIDIデータ編集処理の一例を示すフローチャート図である。まず、この場合には、電子楽器1は、フロッピーディスクドライブ24から読み出したMIDIデータにそのヘッダ情報の一部である電子署名すなわち著作権表示データを分散して記憶すると共にMIDIフォーマットを維持したまま暗号化処理を施す場合について説明する。

【0022】まず、ステップ31では、MIDIデータ列内に分散して書き込まれるべき付属情報の内容を決定する。この実施の形態では、付属情報として、電子署名（著作権表示データ）をMIDIデータ列内に分散して埋め込む場合について説明する。例えば、著作権表示データとして『COPYRIGHT△YMH△1996』のような文字をMIDIデータ列内に分散して書き込む場合には、これらの文字列をパネルスイッチ2Eを用いて入力する。ここで、△は空白を意味する。ステップ31で書き込むべき電子署名すなわち付属情報の文字列が決定したので、今度はステップ32で、その電子署名に関する4ビット構成のデータ列を得る。例えば、パネルスイッチ2Eによって『COPYRIGHT△YMH△1996』が入力された場合には、それをASCIIの文字符号のデータ列に変換する。この場合、それぞれの文字は『C』＝『43H』、『O』＝『4FH』、『P』＝『50H』、『Y』＝『59H』、『R』＝『52H』、『I』＝『49H』、『G』＝『47H』、『H』＝『48H』、『T』＝『54H』、『△』＝『20H』、『Y』＝『59H』、『M』＝『4DH』、『H』＝『48H』、『△』＝『20H』、『1』＝『31H』、『9』＝『39H』、『9』＝『39H』、『6』＝『36H』のような一連のASCIIの4ビット構成のデータで表されるようになる。なお、『36H』のように記された数において、数字の末尾のHはその数が16進表示であることを示す。

【0023】次に、ステップ33では、電子署名として付与する情報すなわち、ステップ32で得られたASCII4ビット構成のデータ列を4ビットレジスタBRに格納する。例えば、『Y』＝『59H』、『M』＝『4DH』、『H』＝『48H』の文字をMIDIデータ列に順番に格納する場合、4ビットレジスタBRには、図1（C）に示すように『0101B』、『1001B』、『0100B』、『1101B』、『0100B』、『1000B』のような4ビット構成のデータが順番に格納される。なお、『1000B』のように記された数において、数字の末尾のBはその数が2進表示であることを示す。そして、ステップ34では、この4ビットレジスタBRに格納されている上位3ビットをノートエリアナンバNAとし、最下位ビット（LSB）をベロシティエリアナンバVAとする。従って、電子署名の『Y』＝『59H』の最初の4ビット：『0101B』の場合はNA＝2，VA＝1、後ろの4ビット：『1001B』の場合はNA＝4，VA＝1となり、電子署名の『M』＝『4DH』の最初の4ビット：『0100B』の場合はNA＝2，VA＝0、後ろの4ビット：『1101B』の場合はNA＝6，VA＝1となり、電子署名の『H』＝『48H』の最初の4ビット：『0100B』の場合はNA＝2，VA＝0、後ろの4ビット：『1000B』の場合はNA＝4，VA＝0となる。

【0024】次に、ステップ35では、MIDIデータ列（Standard MIDI File：SMF）の中からキーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどの各種のMIDIデータを順次取り出す。すなわち、MIDIデータ列は基本的にはキーオンステータスバイト、キーコードバイト、ベロシティバイトからなるキーオンイベントデータやこれ以外のプログラムチェンジイベントデータやコントロールチェンジイベントデータなどから構成されているので、ステップ35では、このようなMIDIデータを順番に取り出す。

【0025】ステップ36では、取り出されたMIDIデータがキーオンイベントデータKONであるかどうかを判定し、キーオンイベントデータ（YES）の場合は次のステップ37以下の処理を行い、そうでない（NO）の場合はステップ3Hに進む。従って、ステップ35で取り出されたMIDIデータがキーオンイベントデータの場合には、図1（A）のようなキーオンイベントデータからなるMIDIデータ列SMF1が得られることになる。このキーオンイベントデータのMIDIデータ列SMF1はデュレーションタイムDと、キーオンイ

ベントデータとの組合せで構成される。ステップ３７では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するキーコードレジスタｂ，ベロシティレジスタｃに格納する。すなわち、キーオンイベントデータのキーオンステータスバイト中のキーコードバイトの値すなわちキーコードをキーコードレジスタｂに、ベロシティバイトの中のベロシティの値をベロシティレジスタｃにそれぞれ格納する。

【００２６】ステップ３８では、前のキーオンイベントデータのキーコードとベロシティとの差分値を取り、それをキーコード差分値レジスタｄｋｅｙ及びベロシティ差分値レジスタｄｖｅｌにそれぞれ格納する。すなわち、キーコードレジスタｂには今回のキーオンイベントデータのキーコードが格納され、前回値レジスタｂ０には前のキーオンイベントデータのキーコードが格納される。キーコードレジスタｂの値から前回値レジスタｂ０の値を減算することによって得られた差分値がキーコード差分値レジスタｄｋｅｙに格納される。ベロシティレジスタｃには今回のキーオンイベントデータのベロシティが格納され、前回値レジスタｃ０には前のキーオンイベントデータのベロシティが格納される。従って、同様にベロシティレジスタｃの値から前回値レジスタｃ０の値を減算することによって得られた差分値がベロシティ差分値レジスタｄｖｅｌに格納される。図１（Ａ）に示されるＭＩＤＩデータ列ＳＭＦ１の場合、このステップ３８の演算によって得られたキーコード差分値レジスタｄｋｅｙ及びベロシティ差分値レジスタｄｖｅｌの値は図１（Ｂ）のようになる。ステップ３９では、次回のステップ３８の処理に備えて、今回のキーコードの値すなわちキーコードレジスタｂの値を前回値レジスタｂ０に格納し、今回のベロシティの値すなわちベロシティレジスタｃの値を前回値レジスタｃ０に格納する。

【００２７】ステップ３Ａでは、キーコード差分値レジスタｄｋｅｙの値が８よりも小さいかどうかを判定し、小さい場合には、次のステップ３Ｂに進み、８以上の場合にはステップ３Ｇに進む。ステップ３Ｂでは、ベロシティ差分値レジスタｄｖｅｌの値が３２よりも小さいかどうかを判定し、小さい場合には、次のステップ３Ｃに進み、３２以上の場合にはステップ３Ｇに進む。すなわち、両方のステップ３Ａ及び３ＢでＮＯと判定された場合には、そのキーオンイベントデータは置換対象外のデータだと判定し、ステップ３Ｇでレジスタｃの最下位ビットを『０』に修正、すなわちベロシティの値を偶数にしてからステップ３Ｈに進む。これによって、ベロシティの値が偶数の場合には、そのキーオンイベントデータが置換対象外のデータであり、修正されなかったことをデータ修復の際に容易に判定することができるようになる。

【００２８】ステップ３Ｃでは、キーコード差分値レジスタｄｋｅｙの値が正又は０の場合と負の場合とで、それぞれ異なる置換演算を行う。すなわち、キーコード差分値レジスタｄｋｅｙの値が正又は０の場合には、ノートエリアナンバＮＡを８倍したものと、キーコード差分値レジスタｄｋｅｙの値と、定数６４との和を求め、その結果を第１キーコードレジスタｂ１に格納する。３ビットからなるノートエリアナンバＮＡの最大値は“７”、最小値は“０”であるから、この条件下では、レジスタｂ１の値の最小値は“６４”である。一方、キーコード差分値レジスタｄｋｅｙの値が負の場合には、定数６３からノートエリアナンバＮＡを８倍したものを減算すると共にキーコード差分値レジスタｄｋｅｙの値を加算し、その結果を第１キーコードレジスタｂ１に格納する。なお、このときキーコード差分値レジスタｄｋｅｙは負の値なので、結果的には定数６３からキーコード差分値レジスタｄｋｅｙの値を減算することになる。この条件下では、レジスタｂ１の値は“６３”よりも大きくなることはない。

【００２９】ステップ３Ｄでは、ベロシティ差分値レジスタｄｖｅｌの値が正又は０の場合と負の場合とで、それぞれ異なる置換演算を行う。すなわち、ベロシティ差分値レジスタｄｖｅｌの値が正又は０の場合には、ベロシティエリアナンバＶＡを３２倍したものと、ベロシティ差分値レジスタｄｖｅｌの値と、定数６４との和を求め、その結果を第１ベロシティレジスタｃ１に格納する。この条件下では、レジスタｃ１の値の最小値は“６４”である。ベロシティ差分値レジスタｄｖｅｌの値が負の場合には、定数６３からベロシティエリアナンバＶＡを３２倍したものを減算すると共にベロシティ差分値レジスタｄｖｅｌの値を加算し、その結果を第１ベロシティレジスタｃ１に格納する。なお、このときベロシティ差分値レジスタｄｖｅｌは負の値なので、結果的には定数６３からベロシティ差分値レジスタｄｖｅｌの値を減算することになる。この条件下では、レジスタｃ１の値は“６３”よりも大きくなることはない。

【００３０】ステップ３Ｅでは、第１キーコードレジスタｂ１の値を新しいキーコードの値とし、第１ベロシティレジスタｃ１の値を新しいベロシティの値としてＭＩＤＩデータを修正する。そして、ステップ３Ｆで、第１ベロシティレジスタｃ１の最下位ビットを『１』に修正、すなわちベロシティの値を奇数に修正してからステップ３Ｈに進む。これによって、ベロシティの値が奇数の場合には、そのキーオンイベントデータが付属情報に基づいて修正されたことをデータ修復の際に容易に判定することができる。ステップ３Ｈでは、全ての付属情報についてステップ３３からステップ３３までの処理が終了したかどうかを判定し、終了している（ＹＥＳ）場合にはこのＭＩＤＩデータ編集処理を終了し、終了していない場合にはステップ３３にリターンし、次の付属情報について一連の処理を行う。

【００３１】例えば、図１の場合には、１番目のキーオ

ンイベントデータ（９１，１００，１００）は、最初のものなので修正されない。２番目のキーオンイベントデータ（９１，１０１，１０２）は、図３のＭＩＤＩデータ編集処理によって、図１（Ｅ）に示すような２番目のキーオンイベントデータ（９１，８１，９９）に変換される。同様にして、３番目のキーオンイベントデータ（９１，１０３，１０２）も図１（Ｄ）に示すような３番目のキーオンイベントデータ（９１，９８，９７）に変換される。そして、４番目のキーオンイベントデータ（９１，１２０，７０）は、図１（Ｂ）に示すように、キーコード差分値レジスタｄｋｅｙの絶対値が『１７』と８以上であるため、ステップ３ＡでＮＯと判定され、置換対象外となり、ステップ３Ｇを経て、キーオンイベントデータ（９１，１２０，７０）がそのまま図１（Ｄ）に示すように４番目のキーオンイベントデータとなる。そして、５番目から８番目までのキーオンイベントデータも同様にして、図１（Ｄ）の５番目から８番目までのキーオンイベントデータのように変換される。なお、この場合に、２番目と３番目のキーオンイベントデータの場合は、ステップ３Ｆの処理によって、ベロシティの値が奇数に変換されている。なお、プログラムチェンジイベントは付属情報の付与されるノートオンイベントなどに比べて、比較的低頻度で発生し、プログラムチェンジ後は、楽器が変わることを意味するので、その位置を始点としてエディットされる可能性の高い単位ということが言えるので、取り出されたＭＩＤＩデータがプログラムチェンジイベントだった場合には、直ちに終了して、付属情報の最初からＭＩＤＩデータを編集するようにしてもよい。

【００３２】図５は、上述の第１の実施の形態に従って得られる各レジスタｂ１，ｃ１の値のとり得る状態を列挙したものである。図５において、縦軸にはレジスタｂ１にストアされるキーコードのとりうる値を割り当て、横軸にはレジスタｃ１にストアされるベロシティデータのとりうる値を割り当て、各値の６３と６４との間を原点とするような直交座標系を構成する。これによって、キーオンイベントデータの中のキーコードが高くなる場合（正の値）又は小さくなる場合（負の値）と、ベロシティが増加する場合（正の値）又は減少する場合（負の値）とからなる４種類の組み合わせがこの直交座標系の各象限に割り当てられることになる。そして、各象限には付属情報となる４ビット構成の１６個のデータが割り当てられる。すなわち、図５に示すようにキーコードが高くなり、ベロシティが増加する場合と、キーコードが高くなり、ベロシティが減少する場合と、キーコードが低くなり、ベロシティが増加する場合と、キーコードが低くなり、ベロシティが減少する場合のそれぞれが各象限に対応するようになる。各象限に割り当てられた４ビット構成の付属情報のうち、最下位ビットは横軸のベロシティに、上位３ビットは縦軸のキーコードに割り当てられる。従って、キーコードが高くなり、ベロシティが増加する場合には、直交座標の第１象限に割り当てられ、キーコード軸の６４～１２７は０００から１１１までの８つの付属情報に対応するように分割され、ベロシティ軸の６４～１２７は０と１の２つの付属情報に対応するように分割される。キーコードが高くなり、ベロシティが減少する場合には、直交座標の第２象限に割り当てられ、キーコード軸の６４～１２７は０００から１１１までの８つの付属情報に対応するように分割され、ベロシティ軸の６３～０は０と１の２つの付属情報に対応するように分割される。キーコードが低くなり、ベロシティが増加する場合には、直交座標の第３象限に割り当てられ、キーコード軸の６３～０は０００から１１１までの８つの付属情報に対応するように分割され、ベロシティ軸の６３～０は０と１の２つの付属情報に対応するように分割される。キーコードが低くなり、ベロシティが減少する場合には、直交座標の第４象限に割り当てられ、キーコード軸の６３～０は０００から１１１までの８つの付属情報に対応するように分割され、ベロシティ軸の６４～１２７は０と１の２つの付属情報に対応するように分割される。

【００３３】この場合、図６に示すように、各付属情報に対応したキーコードには８個の値が存在し、ベロシティには３２個の値が存在するので、ステップ３Ａ及び３Ｂのように、キーコード差分値レジスタｄｋｅｙの値は８よりも小さくなければならず、ベロシティ差分値レジスタｄｖｅｌの値は３２よりも小さくなければならないという条件が入っているのである。従って、キーコードが８よりも小さな値で増加又は減少し、ベロシティが３２よりも小さな値で増加又は減少した場合には、図５の各象限の付属情報が選択され、その付属情報の値、キーコード差分値及びベロシティ差分値に応じて、元のキーコード及びベロシティが編集される。なお、この場合、ベロシティについては、編集された値であるかどうかを示すために、ステップ３Ｇ及びステップ３Ｆによって最下位ビットに処理が加えられ、編集されたベロシティの値は、最終的には奇数値の１６個だけとなる。

【００３４】次に、図３のＭＩＤＩデータ編集処理によって付与された電子情報すなわち電子署名データを検出、すなわち修復するＭＩＤＩデータ修復処理について説明する。図４は、電子楽器１がＭＩＤＩデータ修復装置として動作する場合の一例を示すフローチャート図である。この場合には、フロッピーディスクドライブ２５から読み出されたＭＩＤＩデータに前述のようなヘッダ情報の一部（電子署名すなわち著作権表示データ）が分散記録されているものとする。以下の実施の形態では、この著作権表示データを検出して、それをモニタなどに表示するまでの動作を説明する。まず、ステップ４１では、電子署名を検出すべきＭＩＤＩデータ列、すなわち図３の電子情報付与処理によって電子署名の付与された

ＭＩＤＩデータ列（Ｓｔａｎｄａｒｄ　ＭＩＤＩ　Ｆｉｌｅ：ＳＭＦ）の中から順次キーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどのデータを取り出す。

【００３５】ステップ４２では、取り出されたＭＩＤＩデータがキーオンイベントデータＫＯＮであるかどうかを判定し、キーオンイベントデータ（ＹＥＳ）の場合は次のステップ４３に進み、そうでない（ＮＯ）場合はステップ４Ｃに進む。従って、ステップ４１で取り出されたＭＩＤＩデータがキーオンイベントデータの場合には、図１（Ｆ）のようなキーオンイベントＭＩＤＩデータ列ＳＭＦ２が作成されることになる。ステップ４３では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するキーコードレジスタｂ，ベロシティレジスタｃに格納する。すなわち、キーコードバイトの中のキーコードをキーコードレジスタｂに、ベロシティバイトの中のベロシティをベロシティレジスタｃにそれぞれ格納する。

【００３６】ステップ４４では、ベロシティレジスタｃの最下位ビットが『１』かどうか、すなわち、付属情報（電子署名情報）の付与されたキーオンイベントデータであるのかどうかの判定を行い、ＹＥＳの場合は次のステップ４５に進み、そうでない（ＮＯ）場合はステップ４Ｃに進む。ステップ４５、ステップ４６、ステップ４９及びステップ４Ａでは、相前後するキーオンイベントデータのキーコードレジスタｂ、前回キーコードレジスタｂ０、ベロシティレジスタｃ及び前回ベロシティレジスタｃ０のそれぞれの格納値に基づいて修復キーコードレジスタｂ２、修復ベロシティレジスタｃ２、ノートエリアナンバＮＡ及びベロシティエリアナンバＶＡの値をそれぞれ求める。このとき、キーコードレジスタｂの値が６４以上の場合とこれよりも小さい場合とで、それぞれ異なる演算式を用いて各値を求める。

【００３７】ステップ４５においてはステップ３Ｃ（図３）の置換式の逆算を行なう，すなわち、キーコードレジスタｂの値が６４以上の場合には、キーコードレジスタｂの値から定数６４を減算し、その減算値の８におけるモジュロ（減算値を８で除した余り）を求めることで、ｄｋｅｙを再生する。そして、このｄｋｅｙを前回キーコードレジスタｂ０に加算することでキーコードの修復値を得て、それを修復キーコードレジスタｂ２にストアする。一方、キーコードレジスタｂの値が６４よりも小さい場合には、キーコードレジスタｂの値から定数６３を減算し、その減算値の８におけるモジュロを求めることで、ｄｋｅｙを再生する。そして、このｄｋｅｙを前回キーコードレジスタｂ０に加算することでキーコードの修復値を得て、それを修復キーコードレジスタｂ２にストアする。

【００３８】ステップ４６においてはステップ３Ｄ（図３）の置換式の逆算を行なう。すなわち、ベロシティレジスタｃの値が６４以上の場合には、ベロシティレジスタｃの値から定数６４を減算し、その減算値の３２におけるモジュロ（減算値を３２で除した余り）を求めることで、ｄｖｅｌを再生する。そして、このｄｖｅｌを前回ベロシティレジスタｃ０に加算することでベロシティデータの修復値を得て、それを修復ベロシティレジスタｃ２にストアする。一方、ベロシティレジスタｃの値が６４よりも小さい場合には、ベロシティレジスタｃの値から定数６３を減算し、その減算値の３２におけるモジュロを求めることで、ｄｖｅｌを再生する。そして、このｄｖｅｌを前回ベロシティレジスタｃ０に加算することでベロシティデータの修復値を得て、それを修復ベロシティレジスタｃ２にストアする。

【００３９】ステップ４７において、修復キーコードレジスタｂ２の値をキーコードの値とし、修復ベロシティレジスタｃ２の値をベロシティの値としてＭＩＤＩデータを修復する。ステップ４８では、次回のステップ４５及びステップ４６の処理に備えて、修復されたキーコードの値すなわち修復キーコードレジスタｂ２の値を前回値レジスタｂ０に格納し、修復されたベロシティの値すなわち修復ベロシティレジスタｃ２の値を前回値レジスタｃ０に格納する。ステップ４９においてはステップ３Ｃ（図３）の置換式の逆算を行なうことでＮＡを再生する。すなわち、キーコードレジスタｂの値が６４以上の場合には、キーコードレジスタｂの値から定数６４を減算し、その減算値を８で除した商を求め、それをノートエリアナンバＮＡとする。一方、キーコードレジスタｂの値が６４よりも小さい場合には、定数６３からキーコードレジスタｂの値を減算し、その減算値を８で除した商を求め、それノートエリアナンバＮＡとする。ステップ４Ａではステップ３Ｄ（図３）の置換式の逆算を行なうことでＮＡを再生する。すなわち、ベロシティレジスタｃの値が６４以上の場合には、ベロシティレジスタｃの値から定数６４を減算し、その減算値を３２で除した商を求め、それをベロシティエリアナンバＶＡとする。一方、ベロシティレジスタｃの値が６４よりも小さい場合には、定数６３からベロシティレジスタｃの値を減算し、その減算値を３２で除した商を求め、それベロシティエリアナンバＶＡとする。

【００４０】ステップ４Ｂでは、ノートエリアナンバＮＡを上位３ビットとし、ベロシティエリアナンバＶＡを最下位ビットとする４ビット構成のデータ列からなる付属情報を得る。ステップ４Ｃでは、全ＭＩＤＩデータの修復が終了したかどうかを判定し、終了した場合にはＭＩＤＩデータ修復処理を終了し、そうでない場合にはステップ４１にリターンし、再びＭＩＤＩデータに対する修復処理を行う。以上の一連の修復処理が終了した時点でモニタにそのヘッダ情報の一部（電子署名すなわち著作権表示データ）などを表示するが、ここではそのステップについて省略してある。

【００４１】例えば、図１（Ｅ）のキーオンイベントのＭＩＤＩデータ列ＳＭＦ２と図１（Ｇ）のキーオンイベントのＭＩＤＩデータ列ＳＭＦ２とは同じものであり、図１（Ｇ）のキーオンイベントのＭＩＤＩデータ列ＳＭＦ２に基づいて、図１（Ｆ）のようなＭＩＤＩデータ列ＳＭＦ３が修復され、図１（Ｇ）のような付属情報が抽出される。まず、図１（Ｇ）の１番目のキーオンイベントデータ（９１，１００，１００）は、ベロシティの最下位ビットが『０』すなわち偶数なのでそのままの値がキーオンイベントデータとなる。次に、２番目のキーオンイベントデータ（９１，８１，９９）は、ベロシティの最下位ビットが『１』すなわち奇数であり、６４以上なので、図４のＭＩＤＩデータ修復処理によって、図１（Ｆ）に示すような２番目のキーオンイベントデータ（９１，１０１，１０３）に変換される。同様にして、３番目のキーオンイベントデータ（９１，９８，９７）も図１（Ｆ）に示すような３番目のキーオンイベントデータ（９１，１０３，１０４）に変換される。そして、４番目のキーオンイベントデータ（９１，１２０，７０）は、ベロシティの最下位ビットが『０』すなわち偶数なのでそのままの値がキーオンイベントデータとなる。そして、図１（Ｇ）に示すような５番目から８番目までのキーオンイベントデータも同様にして、図１（Ｆ）の５番目から８番目までのキーオンイベントデータのように変換される。この場合に、２番目と３番目のキーオンイベントデータについては、図３のステップ３Ｆの処理によってベロシティの値が奇数に変換されている関係上、図１（Ｆ）に示されるキーオンイベントデータのベロシティの値がＭＩＤＩデータ編集処理される前の図１（Ａ）のキーオンイベントデータのベロシティの値よりも１～２程度大きくなっている。しかしながら、この程度のベロシティ値の誤差は人間の耳に与える影響は小さいので、ほとんど無視できる範囲の誤差であると言える。また、図１（Ｇ）のキーオンイベントデータに基づいて、図１（Ｈ）のような電子署名情報（著作権表示データ）が再現されることになる。このようにして修復された電子署名情報を図示していない表示装置などで表示する。

【００４２】なお、上述の第１の実施の形態では、ベロシティに１ビット、キーコードに３ビットの付属情報を割り当てる場合について説明したが、これ以外のビットの組み合わせでもよいことはいうまでもない。例えば、ベロシティとキーコードにそれぞれ２ビットの付属情報を割り当ててもよいし、ベロシティのみに４ビットの付属情報を割り当ててもよい。上述の第１の実施の形態では、ノート方向を８等分、ベロシティ方向を２等分して４ビット構成の付加情報を記録する場合について説明したが、ノート方向及びベロシティ方向の分割数を任意に設定することにより、付加情報量を増やすことが可能である。例えば、ノート、ベロシティの両方とも１６等分すれば、１キーオンイベントに８ビット構成の付加情報を付加することもでき、アスキーコードの１バイトならば１イベントのキーオンで表現することも可能となる。なお、この場合には、キーコード差分値レジスタｄｋｅｙ及びベロシティ差分値レジスタｄｖｅｌの値が４以下でないと、そのキーオンイベントには付加情報を割り当てることができなくなるという制限が生じる。従って、予めキーコード差分値及びベロシティ差分値の発生頻度を取得し、それに応じて最も沢山の付属情報を埋め込むことが可能な分割数を選択するようにしてもよい。なお、この場合、暗号化のためのアルゴリズムも、１データ単位に組み込むべき付属情報のドット数に応じて、幾分変更される。例えば、４ビットの付属情報をノートデータに組み込む場合は、図３のステップ３Ｃでの暗号化アルゴリズムの係数“８”が“４”に変更となる。また、上述の第１の実施の形態では、図５に示すような関係を演算式によって形成する場合について説明したが、予め図５のような関係となるようなテーブルを作成しておき、各値をテーブル変換することによって、所定の値が求まるようにしてもよい。この場合には、図５のように付属情報が規則正しく並んでいなくても、ランダムに配列してあってもよいことになる。なお、テーブルは、ＲＯＭやＲＡＭのようなメモリ素子で構成するものに限らず、ロジックゲート回路で構成してもよいし、ソフトウェアプログラムで構成してもよい。

【００４３】上述の第１の実施の形態では、フロッピーディスクドライブ２４から読み出されたＭＩＤＩデータにおいてそのヘッダ情報の一部である電子署名すなわち著作権表示データを比較的大きなデータ単位でキーオンイベントデータ中に分散して記憶すると共にＭＩＤＩフォーマットを維持したまま暗号化処理を施す（電子透しを施す）という場合について説明した。これだと、キーオンイベントデータの中にはそのキーコード及びベロシティの内容がほとんど書き換えられるものが出てくるので、所定のＭＩＤＩデータ修復処理を行わずにＭＩＤＩデータ再生処理を行っても再生される楽音は、全くデタラメなものとなってしまう。この点に鑑みて、次に説明する第２の実施の形態では、上記とは異なるやり方で「電子透し」を施すようにしている。この第２の実施の形態では、ＭＩＤＩデータの最初の部分に、複数の暗号化処理の中のどの暗号化処理が施されたのかを示すスクランブルデコードデータを埋め込み、そこには暗号化処理を行わずに、通常のＭＩＤＩデータの再生を行えるようにし、残りのＭＩＤＩデータ部分にスクランブルデコードデータに対応したスクランブル処理を行い、ＭＩＤＩデータの再生を行うことができないようにしている。

【００４４】図７は、ＭＩＤＩデータ中に埋め込まれるスクランブルデコードデータの内容を示す図である。このスクランブルデコードデータは２バイト構成のデータであり、第１バイト目の上位４ビットはそのアルゴリズ

ムを示し、下位４ビットはスクランブル処理されるイベント数すなわちカウント数を示し、第２バイト目の８ビットはそのアルゴリズムに対応したバリューを示す。ここで、アルゴリズムの種類としては、キーオンデータのノート番号を変更するというもの、キーオンデータのノートとベロシティを入れ替えるというもの、インターバルデータを変更するというもの、キーオンデータのチャンネルデータを変更するというものなどがある。カウント数は、前述のようにＡＳＣＩＩの２文字分すなわち１６ビット分のデータを作成するのに必要な数のキーオンイベントデータの先頭からいくつに対してスクランブル処理を行うのかを示すものである。

【００４５】上述の第１の実施の形態の場合には、１つのキーオンイベントデータに４ビット分のデータを埋め込むことができたので、２つのキーオンイベントデータにＡＳＣＩＩの１文字分の付属情報を埋め込むことができたが、今回の第２の実施の形態では、１つのキーオンイベントデータに１ビット分のデータしか埋め込めない場合について説明するので、修正するイベント数の最大値は１６となる。バリューは対応するアルゴリズムによって異なるものであり，例えば、キーオンデータのノート番号を変更するというアルゴリズムの場合には、その変更する度合いすなわちトランスポーズ値（修正ノート値）であり、インターバルデータを変更するというアルゴリズムの場合には、その変更する修正値であり、キーオンデータのチャンネルデータを変更する場合には、その修正チャンネル値である。このように全部で１６種類のアルゴリズムが規定されているものする。なお、この１６種類のアルゴリズムの中の一つとして、前述の第１の実施の形態に係るようなスクランブル処理が存在してもよいことはいうまでもない。

【００４６】図８は、電子楽器１が電子情報処理システムとして動作する場合のＭＩＤＩ編集処理の別の一例となるＭＩＤＩ編集処理２を示すフローチャート図である。まず、この場合には、電子楽器１は、フロッピーディスクドライブ２４から読み出したＭＩＤＩデータにそのヘッダ情報の一部である電子署名すなわち著作権表示データを分散して記憶すると共に前述の１６ビット構成のスクランブルデコードデータを記憶する。まず、ステップ８１では、ＭＩＤＩデータ列内に分散して書き込まれるべきスクランブルの内容を決定する。すなわち、図８のどのアルゴリズムに基づいてスクランブル処理を行ううのかがパネルスイッチ２Ｅによって予め設定されるものとする。スクランブルの内容がステップ８１で決定したので、今度はステップ８２でそのスクランブルのデコードデータ列を得る。例えば、パネルスイッチ２Ｅによって、第５番目のアルゴリズムが選択された場合には、そのアルゴリズムの種類を示すデータ『０１０１Ｂ』と、そのアルゴリズムによって修正されるカウント数『１００１Ｂ』とからなる８ビット構成のデータ列がスクランブルデコードデータ列としてＭＩＤＩデータ列内に分散して記録されるようになる。

【００４７】次に、ステップ８３では、ステップ８２で得られたスクランブルデコードデータ列の１バイト分をバイトレジスタＢＲに格納する。そして、ステップ８４では、このバイトレジスタＢＲに格納されたスクランブルデコードデータの各ビットを反転して、別の８ビット構成のビット列を作成する。例えば、スクランブルデコードデータ列が図１２（Ｆ）に示すように『０１０１１００１Ｂ』のような場合には、ステップ８４のデータ変換すなわちビット反転によって、図１２（Ｅ）のようなビット列『１０１００１１０Ｂ』になる。次に、ステップ８５では、ＭＩＤＩデータ列（Ｓｔａｎｄａｒｄ ＭＩＤＩＦｉｌｅ：ＳＭＦ）の中からキーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどの各種のＭＩＤＩデータを順次取り出す。すなわち、ＭＩＤＩデータ列は基本的にはキーオンステータスバイト、キーコードバイト、ベロシティバイトからなるキーオンイベントデータやこれ以外のプログラムチェンジイベントデータやコントロールチェンジイベントデータなどから構成されているので、ステップ８５では、このようなＭＩＤＩデータを順番に取り出す。

【００４８】ステップ８６では、取り出されたＭＩＤＩデータがキーオンイベントデータＫＯＮであるかどうかを判定し、キーオンイベントデータ（ＹＥＳ）の場合は次のステップ８７に進み、そうでない（ＮＯ）の場合はステップ８Ｄに進む。従って、ステップ８５で取り出されたＭＩＤＩデータがキーオンイベントデータの場合には、図１２（Ａ）のようなキーオンイベントデータからなるＭＩＤＩデータ列ＳＭＦ１が得られることになる。このキーオンイベントデータのＭＩＤＩデータ列ＳＭＦ１はデュレーションタイムＤと、キーオンイベントデータとの組合せで構成される。ステップ８７では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するレジスタａ，ｂ，ｃに格納する。すなわち、キーオンイベントデータの中のキーオンステータスバイトの中のチャンネル番号をチャンネルレジスタａに、次の第１のデータバイト（キーコードバイト）の中のキーコードをキーコードレジスタｂに、第２のデータバイト（ベロシティバイト）の中のベロシティをベロシティレジスタｃにそれぞれ格納する。

【００４９】ステップ８８では、ステップ８４で作成されたビット列の先頭から１ビット取り出して、それを第１のビットフラグＢＦ１に格納する。次のステップ８９では、ステップ８７の各レジスタａ，ｂ，ｃの格納値を所定の関数に従って演算し、その演算結果によって得られたビットデータを反転して、第２のビットフラグＢＦ２に格納する。この実施の形態では、各レジスタａ，ｂ，ｃの値の合計値のモジュロ２を所定の関数とする。すなわち、関数ｆ１（ａ，ｂ，ｃ，）＝（ａ＋ｂ＋ｃ）

ｍｏｄ２とする。例えば、図１２（Ａ）のようなキーオンイベントＭＩＤＩデータ列ＳＭＦ１の場合には、８つのキーオンイベントデータでスクランブルデコードデータのアルゴリズム及びカウント数に相当するデータが作成される。まず、各キーオンイベントデータの所定の関数ｆ１（ａ，ｂ，ｃ）で演算すると、図１２（Ｂ）のようになる。すなわち、（ａ＋ｂ＋ｃ）の値が偶数なら『０』、奇数なら『１』となる。そして、これらの反転ビットが第２のビットフラグＢＦ２に格納され、図１２（Ｃ）のようになる。

【００５０】ステップ８Ａでは、ステップ８８及びステップ８９で得られた第１のビットフラグＢＦ１と第２のビットフラグＢＦ２とが等しいかどうかを判定し、等しい（ＹＥＳ）場合には次のステップ８Ｂ及びステップ８Ｃの処理を行い、等しくない（ＮＯ）場合はステップ８Ｅにジャンプする。ステップ８Ｂでは、第１のビットフラグＢＦ１と第２のビットフラグＢＦ２とが等しいと判定されたので、ベロシティレジスタｃの値すなわちベロシティデータの最下位ビットに『１』を加算する。そして、ステップ８Ｃで、そのレジスタｃの値に基づいてＭＩＤＩデータ列中のキーオンイベントデータを構成するベロシティの値を更新する。すなわち、ベロシティの値を１だけ増加する。例えば、図１２の場合には、第１のビットフラグＢＦ１と第２のビットフラグＢＦ２との３番目、５番目、６番目及び７番目がそれぞれ等しいと、ステップ８Ａで判定されるので、ＭＩＤＩデータ列中の３番目、５番目、６番目及び７番目のキーオンイベントデータのベロシティの値が『１』だけ増加される。すなわち、３番目のキーオンイベントデータのベロシティ値『６３』が『６４』になり、５番目のキーオンイベントデータのベロシティ値『７８』が『７９』になり、６番目のキーオンイベントデータのベロシティ値『９１』が『９２』になり、７番目のキーオンイベントデータのベロシティ値『４２』が『４３』になる。

【００５１】ステップ８Ｄでは、ステップ８６又はステップ８ＡでＮＯと判定された場合又はステップ８Ｃの処理を終えた場合に行われる処理であって、ステップ８５～ステップ８Ｃまでの処理を１巡回処理とした場合にこの１巡回処理が８回分すなわち８ビット分行われたかどうかを判定し、８回分行われた（ＹＥＳ）場合には、次のステップ８Ｅに進み、そうでない（ＮＯ）の場合は、ステップ８５にリターンし、次の１ビットに対してステップ８５～ステップ８Ｃの処理を行う。ステップ８Ｅでは、ステップ８３～ステップ８Ｄまでの処理を１巡回処理とした場合にその１巡回処理がスクランブルデコードデータの構成バイト数だけ行われたかどうかを判定し、行われた（ＹＥＳ）場合には、次のステップ８Ｆに進み、そうでない（ＮＯ）場合は、ステップ８３にリターンし、スクランブルデコードデータの次の１バイトのバリューに対してステップ８３～ステップ８Ｄの処理を行う。

【００５２】ステップ８Ｆでは、ステップ８１で選択されたスクランブルの内容に応じて、ＭＩＤＩデータ列の後半部分にスクランブル処理を施す。例えば、前述のスクランブルデコードデータの埋め込み終了後のキーオンデータに対してスクランブル処理を行うようにしてもよいし、スクランブルデコードデータの埋め込み終了後、所定数のキーオンデータ経過後にスクランブル処理を行うようにしてもよい。また、スクランブルデコードデータの埋め込みを複数回行い、その後にスクランブル処理を行うようにしてもよい。この場合に、取り出されたＭＩＤＩデータがプログラムチェンジイベントＰＣＭの場合には、スクランブルの内容をランダムに変更して、スクランブルデコードデータの埋め込み処理及びスクランブルの内容に応じたＭＩＤＩデータ列の編集処理を行うようにしてもよい。プログラムチェンジイベントはノートオンイベントなどに比べて、比較的低頻度で発生し、プログラムチェンジ後は、楽器が変わることを意味するので、その位置を始点としてスクランブルの内容を変更し、そのスクランブルデコードデータを埋め込み、スクランブル処理を行うことによって、一定の楽器についてはスクランブルの掛からない演奏を行うことができ、ある所定時間経過後に演奏にスクランブルをかけることができる。

【００５３】次に、図８のＭＩＤＩ編集処理２によって付与されたスクランブルデコードデータの修復処理について説明する。なお、スクランブル処理の解除については、アルゴリズムに応じて逆の処理を施せばいいので、ここではその説明は省略する。図９は、電子楽器１がＭＩＤＩデータ修復装置して動作する場合の一例となるＭＩＤＩデータ修復処理２を示すフローチャート図である。この場合には、フロッピーディスクドライブ２４から読み出されたＭＩＤＩデータのベロシティバイト部分にそのヘッダ情報の一部である電子署名すなわち著作権表示データが分散記録されているとともにスクランブルデコードデータが記録されているものとする。以下の実施の形態では、このスクランブールデコードデータを検出して、それに基づいてＭＩＤＩデータを修復するまでの動作について説明する。

【００５４】まず、ステップ９１では、デコードデータ格納レジスタＤＥＣＯＤＥにナルデータを格納する。すなわち、デコードデータ格納レジスタＤＥＣＯＤＥの内容をリセットする。ステップ９２では、スクランブルデコードデータを検出すべきＭＩＤＩデータ列、すなわち図８のＭＩＤＩデータ編集によってスクランブルデコードデータの付与されたＭＩＤＩデータ列を取り出す。そして、次のステップ９３で、バイトレジスタＢＲとビットカウンタＢＣＮの値をリセットする。

【００５５】次に、ステップ９４では、ＭＩＤＩデータ列（Ｓｔａｎｄａｒｄ　ＭＩＤＩＦｉｌｅ：ＳＭＦ）の

中から順次キーオンイベントデータやプログラムチェンジデータやコントロールチェンジデータなどのデータを取り出す。ステップ９５では、取り出されたＭＩＤＩデータがキーオンイベントデータＫＯＮであるかどうかを判定し、キーオンイベントデータ（ＹＥＳ）の場合は次のステップ９６に進み、そうでない（ＮＯ）場合はステップ９Ｃに進む。従って、ステップ９４で取り出されたＭＩＤＩデータがキーオンイベントデータの場合には、図１２（Ｇ）のようなキーオンイベントＭＩＤＩデータ列ＳＭＦ２が作成されることになる。なお、図１２（Ｇ）のキーオンイベントデータ列は先に電子署名の付与された図１２（Ｄ）のキーオンイベントデータ列ＳＭＦ２と同じものである。

【００５６】ステップ９６では、キーオンイベントデータの中の各バイトのデータをそれぞれ対応するレジスタａ，ｂ，ｃに格納する。すなわち、キーオンイベントデータの中のキーオンステータスバイトの中のチャンネル番号をチャンネルレジスタａに、次の第１のデータバイト（キーコードバイト）の中のキーコードをキーコードレジスタｂに、第２のデータバイト（ベロシティバイト）の中のベロシティをベロシティレジスタｃにそれぞれ格納する。ステップ９７では、バイトレジスタＢＲの値を２倍、すなわち、シフトして、その最下位ビットに関数ｆ１（ａ，ｂ，ｃ）＝（ａ＋ｂ＋ｃ）ｍｏｄ２の反転値を加算する。ステップ９８では、ビットカウンタＢＣＮの値を１だけインクリメントする。ステップ９９では、ビットカウンタＢＣＮの値が『８』になったかどうかを判定し、ＹＥＳの場合は次のステップ９Ａに進み、ＮＯの場合はステップ９Ｃにジャンプする。

【００５７】例えば、図１２（Ｇ）のようなキーオンイベントのＭＩＤＩデータ列ＳＭＦ２の場合には、８つのキーオンイベントデータで１バイト分のスクランブルデコードデータが作成される。まず、各キーオンイベントデータの所定の関数ｆ１（ａ，ｂ，ｃ）で演算すると、図１２（Ｈ）のようになる。すなわち、（ａ＋ｂ＋ｃ）の値が偶数なら『０』、奇数なら『１』となる。そして、これらの反転ビットは図１２（Ｊ）のようになり、その値がバイトレジスタＢＲを順次シフトしていくため、最終的には図１２（Ｊ）のような１バイト分のスクランブルデコードデータが形成される。このようにして、得られた図１２（Ｊ）のデータは図１２（Ｆ）と同じである。すなわち、ＭＩＤＩデータのベロシティバイトに分散して書き込まれたスクランブルデコードデータ、すなわち図１２（Ｆ）のようなデータが図１２（Ｊ）のように忠実に再現されたことになる。

【００５８】ステップ９Ａでは、バイトレジスタＢＲの格納値をデコードデータ格納レジスタＤＥＣＯＤＥに追加記憶する。ステップ９Ｂで、バイトレジスタＢＲとビットカウンタＢＣＮの値をリセットする。ステップ９Ｃでは、ステップ９５でＮＯと判定された場合、又はステップ９Ｂの処理を終了した場合に行われる処理であって、ステップ９４～ステップ９Ｂの処理がスクランブルデコードデータに対応する２バイト分終了したかどうかの判定を行い、この判定結果が２バイト分終了（ＹＥＳ）の場合には、次のステップ９Ｄに進み、終了していない（ＮＯ）場合にはステップ９４にリターンし、一連の処理を繰り返し実行する。ステップ９Ｄでは、スクランブルデコードデータの内容に基づいてスクランブル処理されたＭＩＤＩデータの修復処理を行う。

【００５９】なお、上述の実施の形態では、ＭＩＤＩデータ列からキーオンイベントデータを順番に取り出して、それに対して順番にスクランブルデコードデータを格納する場合について説明したが、これだと、１つのＭＩＤＩデータ列に対して１種類のスクランブル処理しか行えない。そこで、チャンネル単位毎に異なるスクランブル処理を行えるようにした第３の実施の形態について説明する。図１０は、電子楽器１がＭＩＤＩデータ編集装置して動作する場合の一例となるＭＩＤＩデータ編集処理３を示すフローチャート図である。まず、この場合は、電子楽器１は、フロッピーディスクドライブ２４から読み出したＭＩＤＩデータ中に前述のようなスクランブルデコードデータと共にヘッダ情報の一部である電子署名すなわち著作権表示データを併せて分散記録する。

【００６０】まず、ステップ１０１では、ＭＩＤＩデータ列内に分散して書き込まれるべき電子署名（著作権表示データ）の内容を決定するとともに、各チャンネルに対応したスクランブルデコードデータの内容を決定する。例えば、著作権表示データとして『ＣＯＰＹＲＩＧＨＴ△ＹＭＨ△１９９６』のような文字をＭＩＤＩデータ列内に分散して書き込む場合には、これらの文字列をパネルスイッチ２Ｅを用いて決定する。ここで、△は空白を意味するものとする。書き込むべき電子署名すなわち文字列及びスクランブルデコードデータの内容が決定したので、今度はステップ３２で、その電子署名及びスクランブルデコードデータに関するデータ列を得る。例えば、パネルスイッチ２Ｅによって『ＣＯＰＹＲＩＧＨＴ△ＹＭＨ△１９９６』が入力された場合には、それをＡＳＣＩＩの文字符号のデータ列に変換する。この場合は、『Ｃ』＝『４３Ｈ』、『Ｏ』＝『４ＦＨ』、『Ｐ』＝『５０Ｈ』、『Ｙ』＝『５９Ｈ』、『Ｒ』＝『５２Ｈ』、『Ｉ』＝『４９Ｈ』、『Ｇ』＝『４７Ｈ』、『Ｈ』＝『４８Ｈ』、『Ｔ』＝『５４Ｈ』、『△』＝『２０Ｈ』、『Ｙ』＝『５９Ｈ』、『Ｍ』＝『４ＤＨ』、『Ｈ』＝『４８Ｈ』、『△』＝『２０Ｈ』、『１』＝『３１Ｈ』、『９』＝『３９Ｈ』、『９』＝『３９Ｈ』、『６』＝『３６Ｈ』からなる一連のＡＳＣＩＩのデータ列が得られることになる。

【００６１】これらの文字符号のデータ列は前述の第２の実施の形態において説明した、スクランブルデコードデータと同様の手法でＭＩＤＩデータの中に埋め込むこ

とができることはいうまでもない。従って、以下の処理では、これらのＡＳＣＩＩのデータ列をスクランブルデコードデータ列の埋め込み法と同様のやり方で、ＭＩＤＩデータ列中に分散記録するものとする。すなわち、次のステップ１０３では、図８のステップ８３～ステップ８Ｆと同様の処理をそのチャンネルに対してそれぞれ行う。このとき、ステップ８９の所定の関数に従った演算処理をそのチャンネルに対してそれぞれ異ならせて行う。例えば、ステップ８９では、関数ｆ１（ａ，ｂ，ｃ，）＝（ａ＋ｂ＋ｃ）ｍｏｄ２を第２のビットフラグＢＦ２に格納しており、これを第１の関数処理とする。そして、この第１の関数処理に加えて、次の第２、第３及び第４の関数処理の中から、適当なものを各チャンネル毎に使い分けるようにする。第２の関数処理は、ｆ２（ａ，ｂ，ｃ）＝（ａ＋ｃ）ｍｏｄ２、第３の関数処理は、ｆ３（ａ，ｂ，ｃ）＝（ｂ＋ｃ）ｍｏｄ２、第４の関数処理は、ｆ４（ａ，ｂ，ｃ）＝（ｃ）ｍｏｄ２、である。

【００６２】ステップ１０３では、これらの第１から第４までの関数処理をＭＩＤＩチャンネル毎に適宜選択的に切り換えて、図８のステップ８３～ステップ８Ｆと同様の処理を行い、それぞれのチャンネル毎にその選択された関数処理に従って変換された電子署名データ及びスクランブルデコードデータをＭＩＤＩデータ中に冗長的に組み込む。このように、チャンネル毎に関数処理の内容を変更することによって、或るチャンネルのデータを一度に変更するようなエディット処理、例えばチャンネル情報の入れ替えやキーコードのシフトなどといったものが行われた場合でも、エディットによって変更される特定のチャンネルのデータに関係しない関数を利用している別のチャネルのデータから電子署名データやスクランブルデコードデータを復元することができる。

【００６３】次に、図１０のＭＩＤＩデータ編集処理３によって編集されたＭＩＤＩデータを修復するＭＩＤＩデータ修復処理３について図１１を用いて説明する。図１１は、電子楽器１がＭＩＤＩデータ修復装置として動作する場合の一例であるＭＩＤＩデータ修復処理３を示すフローチャート図である。この場合、フロッピーディスクドライブ２４から読み出されたＭＩＤＩデータの各チャンネル毎に所定の関数処理（前述の第１から第４までのいずれかの関数処理）によって、ベロシティバイト部分にそのヘッダ情報の一部である電子署名（著作権表示データ）及びスクランブルデコードデータが分散記録されているものとする。従って、以下の実施の形態では、分散記録された著作権表示データをチャンネル毎に所定の関数処理によって検出して、それをモニタなどに表示したり、スクランブルデコードデータに基づいてＭＩＤＩデータを修復するという動作を行う場合について説明する。

【００６４】まず、ステップ１１１では、電子署名を検出すべきＭＩＤＩデータ列、すなわち図１０のＭＩＤＩデータ編集処理３によって各チャンネル毎に所定の関数処理に応じて電子署名データ及びスクランブルデコードデータの付与されたＭＩＤＩデータ列を取り出す。そして、次のステップ１１２で、チャンネルカウンタをリセットし、ステップ１１３で関数カウンタをリセットする。このチャンネルカウンタと関数カウンタは、後の処理で、各チャンネルに対して各関数処理を施すために利用されるものである。例えば、ＭＩＤＩチャンネルが１６チャンネル相当の場合には、チャンネルカウンタは０から１５までを巡回的にカウントするように処理される。また、関数カウンタは、前述のように関数処理が４種類の場合には、０～３まで巡回的にカウントするように処理される。関数カウンタが『０』の場合は第１の関数処理ｆ１（ａ，ｂ，ｃ，）＝（ａ＋ｂ＋ｃ）ｍｏｄ２を、『１』の場合は第２の関数処理ｆ２（ａ，ｂ，ｃ）＝（ａ＋ｃ）ｍｏｄ２を、『２』の場合は第３の関数処理ｆ３（ａ，ｂ，ｃ）＝（ｂ＋ｃ）ｍｏｄ２を、『３』の場合は第４の関数処理ｆ４（ａ，ｂ，ｃ）＝（ｃ）ｍｏｄ２を意味する。

【００６５】ステップ１１４では、チャンネルカウンタの値に該当するチャンネルについて、関数カウンタの値に該当する関数処理を用いて図９のステップ９１からステップ９Ｃまでと同様にして電子署名データ及びスクランブルデコードデータの検出処理を行う。ステップ１１５では、検出された電子署名データ列に有意な部分があるかどうかの判定を行う。ここで有意な部分とは、ＡＳＣＩＩの文字データによって構成される『ＣＯＰＹＲＩＧＨＴ』なる署名データの一部分である。従って、ステップ１１５では、図９の処理によってデコードデータ格納レジスタＤＥＣＯＤＥに順次追加されたＡＳＣＩＩの文字データによって構成された文字列の中に『ＣＯＰＹＲＩＧＨＴ』なる文字列データが存在するかどうかを判定することになる。ステップ１１５の判定の結果、デコードデータ格納レジスタＤＥＣＯＤＥ内に有意な部分が存在しない（ＮＯ）と判定された場合には、次のステップ１１６に進み、関数カウンタを１だけインクリメントする。そして、次のステップ１１７で、全ての関数処理に対してステップ１１４及びステップ１１５の処理すなわち署名データ及びスクランブルデコードデータの抽出処理が行われたかどうかの判定を行い、行われた（ＹＥＳ）場合には次のステップ１１８に進み、そうでない（ＮＯ）場合はステップ１１４にリターンし、次の関数処理についてステップ１１４及びステップ１１５の処理を行う。逆に、ステップ１１５の判定の結果、デコードデータ格納レジスタＤＥＣＯＤＥ内に有意な部分が存在した場合には、ステップ１１９に進み、スクランブルデコードデータの内容に基づいてＭＩＤＩデータの修復を行う。すなわち、ステップ１１５における判定がＹＥＳであったということは、検出されたスクランブルデコー

ドデータも有意なものだと判断できるので、そのスクランブルデコードデータに基づいて元のＭＩＤＩデータを修復する。なお、検出された電子署名データをモニタ上に表示するようにしてもよいことはいうまでもない。

【００６６】ステップ１１７でＹＥＳと判定されたということは、全てのＭＩＤＩチャンネルに対して、全ての関数処理を用いて電子署名データ及びスクランブルデコードデータの抽出処理を行った結果、データ列に有意な部分が存在しなかったということだから、ステップ１１８では、その結果として『電子署名データ及びスクランブルデコードデータの検出失敗』などの文字をモニタ上に表示する。ステップ１１Ａでは、チャンネルカウンタを１だけインクリメントする。そして、次のステップ１１Ｂで全てのチャンネルに対してステップ１１４及びステップ１１５の処理すなわち電子署名データ及びスクランブルデコードデータの抽出処理が行われたかどうかの判定を行い、行われた（ＹＥＳ）場合には処理を終了し、そうでない（ＮＯ）場合はステップ１１３にリターンし、次のＭＩＤＩチャンネルについてステップ１１３～ステップ１１５の処理を行う。

【００６７】なお、ＭＩＤＩデータに基づく楽音再生演奏において、部分的に変更されたベロシティデータがそのまま演奏に利用されるようにしてよい。これは、ベロシティデータの最下位ビットの値が１だけ増加したとしても、楽音再生精度にはさほどの悪影響を与えないからである。勿論、電子署名データの組込みによって引き起こされたどんな誤差でもベロシティデータから取り除くように、部分的に変更されたベロシティデータを正確な値で再生するようにすることも可能である。なお、上述の実施の形態では、署名データやスクランブルデコードデータをＭＩＤＩデータのベロシティ部分に分散記録する場合について説明したが、これは一例であり、これ以外の著作者名、曲の題名、画像／映像の題名などに関する文字データやデータ形式に関するデータやその他の種々のデータを分散記録してもよいことはいうまでもない。

【００６８】また、ベロシティ部分以外の、デュレーションタイム（待ち時間）データに分散記録してもよいし、その両方に分散記録するようにしてもよい。両方に分散記録する場合、その厳密性すなわち分散記録することによってそのデータの内容が変更する度合いに応じて、分散記録する箇所を適宜切り換えるようにしてもよい。すなわち、デュレーションタイムデータの短い領域で、そのデュレーションタイムデータに分散記録すると、それによる変動の割合（厳密性）が大きくなるので、好ましくないが、デュレーションタイムデータが比較的大きい領域では、逆の関係にあるので、分散記録しても影響は少ない。同じく、ベロシティ部分に記録する場合でも、ベロシティの値の小さい領域で、そのベロシティ部分に分散記録すると、それによる変動の割合（厳密性）が大きくなるので、好ましくないが、ベロシティの値の大きい領域では、分散記録しても影響は少ない。従って、これらを適宜考慮して所定値よりも大きいベロシティに種々のデータを分散記録するようにしてもよい。

【００６９】また、上述の実施の形態では、ＭＩＤＩデータに分散記録する場合について説明したが、これ以外の波形データやシーケンスデータ、ディジタル記録音声データ、画像データ、動画データ、レジストレーション（電子楽器の設定記録）データなどに分散記録するようにしてもよい。波形データに分散記録する場合には、所定の関数処理としてその波形データ（ｗａｖｅ＿ｄａｔａ）そのもののモジュロ２、すなわち、ｆ１（ｗａｖｅ＿ｄａｔａ）＝ｗａｖｅ＿ｄａｔａ　ｍｏｄ２としてもよい。また、波形データの値とそのポイントデータとを用いて、両データの和のモジュロ２、すなわち、ｆ２（ｗａｖｅ＿ｄａｔａ，ｓａｍｐｌｅ＿ｐｏｉｎｔ）＝（ｗａｖｅ＿ｄａｔａ＋ｓａｍｐｌｅ＿ｐｏｉｎｔ）ｍｏｄ２としてもよい。図１０及び図１１の実施の形態では、各（論理）チャンネル毎に関数処理の種類を変更して、署名データを検出する場合について説明したが、プログラムチェンジデータの検出タイミング毎に関数処理の種類を変更するようにしてもよい。この場合には、検出されたデータ列に有意な部分が存在するかどうかの判定処理も各プログラムチェンジデータの検出タイミング毎に行う必要がある。

【００７０】上述の実施の形態では、検出された電子署名データをモニタ上に表示する場合について説明したが、検出側において、電子署名データが存在しない場合や電子署名データが完全に復旧できない場合（各論理チャンネルにおいて、電子署名データが検出できるチャンネルと検出できないチャンネルが混在する場合）には、データの再生を中止するようにしてもよい。また、その検出側の電子楽器やコンピュータなどがネットワークに接続されている場合には、そのネットワーク上に不正にエディットされたデータが存在することを送信し、それをホストコンピュータ側で検出することができるようにしてもよい。上述の実施の形態では、電子楽器がＭＩＤＩデータ編集装置又はＭＩＤＩデータ修復装置として動作する場合について説明したが、同様の処理を行うハードウェアを別途構成するようにしてもよいし、ソフトウェア、ＤＳＰとマイクロプログラムなどで構成するようにしてもよい。また、このようなＭＩＤＩデータ編集装置又はＭＩＤＩデータ修復装置をフロッピーディスクドライブや通信インターフェイスなどに予め内蔵しておいて、データの入出力の段階で強制的に電子署名データを付与したり、検出したりするようにしてもよい。

【００７１】上述の実施の形態では、電子署名データやスクランブルデコードデータを付与する場合を説明したが、電子署名データやスクランブルデコードデータの付

与されたデータを、フロッピーディスクやコンパクトディスクなどで供給するようにしてもよいし、ネットワークを通じて電子的に供給するようなデータ形態を採用してもよい。図8のステップ8Bでは、ベロシティデータの値を1だけインクリメント処理する場合について説明したが、デクリメント処理してもよいし、これらの処理を適当なタイミング（所定データ数毎とか）で交互に行うようにしてもよい。上述の実施の形態では、所定の関数処理の後にビット反転処理を行う場合について説明したが、これは行わなくてもよい。また、ビット反転処理の他にも、上位ビットと下位ビットを入れ替えるとか、AND、OR、XORなどの論理演算処理を施すとか、種々の処理を加えるようにしてもよいことはいうまでもない。また、適宜の暗号化処理を施すようにしてもよい。例えば、上述の第1から第4までの関数処理の結果を適宜組み合わせてもよいし、前回の関数処理の結果を次の関数処理に組み合わせて演算するようにしてもよい。なお、これらの各処理はある程度高速に行うことができ、変換後、逆変換によりもとに戻せるものであれば、これ以外の方法でもよいことはいうまでもない。

【0072】上述の実施の形態では、プログラムチェンジデータの検出タイミング毎に新たに電子署名データやスクランブルデコードデータの付与処理を行う場合について説明したが、これに限らず、小節単位（例えば8小節毎など）データの検出タイミング毎に行うようにしてもよい。また、波形データに分散記録する場合には、所定のサンプル数毎や所定のゼロクロス毎に新たに署名データの付与処理を行うようにしてもよい。ゼロクロス（波形の値の符号が変わるタイミング）は波形エディットの際の基準地点と考えられるからである。また、上述の実施の形態では、電子署名データや題名などのような付属情報を記録する場合について説明したが、MIDIデータと波形データを合わせて記録する場合に、MIDIデータにその波形データの一部を記録するようにしてもよいし、波形データにMIDIデータの一部を合わせて記録するようにしてもよい。

【0073】なお、上述の実施の形態では、署名データをMIDIデータのベロシティ部分に分散記録する場合について説明したが、これは一例であり、これ以外の著作者名、曲の題名（曲名）、画像／映像の題名などに関するテキスト情報や、そのデータ形式に関する情報やその他の種々の電子情報（曲の解説、著作者に関する情報、著作年、歌詞、ニュース、ハードID（機種名、使用OS）などの各種情報）を分散記録してもよいことはいうまでもない。この場合には、図8から図11までの各処理における電子署名をこれらの各種電子情報に置き換えて処理すればよい。このようにして処理された電子情報をモニタ画面上に表示するようにしてもよい。なお、このとき、付属情報を一連の纏まったファイル情報として検出して、これをメモリ等に記憶するあるいは適宜これを利用する、ようにしてもよいし、自動演奏中にMIDIデータから検出される電子情報の流れとして認識し、それをリアルタイムに画面表示したりしてもよい。この場合、付属情報が画像情報の場合には、主要情報であるMIDIデータの受信に伴ってリアルタイムに画像が描画されるようになり、テキスト情報の場合には、ニュース配信などのようなリアルタイムな情報表示を行うことが可能となる。また、情報を付加するための技術がシンプルであるため、オンデマンドなどで電子情報を埋め込むことができるので、TV放送などのようなリアルタイムな描画が可能となる。また、通信ネットワークを通した情報通信において本発明を適用してもよく、例えば、インターネットや電子メールなどのIDやパスワードの配信にも利用することができる。また、簡単なサウンドメッセージを付属情報としてもよい。

【0074】上記のような、通信ネットワークを通じた本発明の実施の形態を略示するシステム構成図が、図16に示されている。送信側において、送信しようとする主要情報（例えばMIDI情報のような音楽演奏情報）がサーバー160から供給される。スクランブルエンコーダ161では、本発明の各実施例に示したやり方で、この主要情報に対して任意の付属情報がリアルタイムで組み込まれる。付属情報を適宜組み込んだ主要情報がネットワーク163を介して送信される。受信側では、デコーダ164において、本発明の各実施例に示したやり方で、受信したデータから付属情報が組み込まれている部分を検出し、これをデコードして、付属情報と主要情報を分離して再生する。MIDI情報のような音楽演奏情報からなる主要情報は音源165に与えられて、音楽演奏を再生する。テキストや静止画像あるいは動画等からなる付属情報はディスプレイ166に与えられて、リアルタイムで表示される。スクランブルエンコーダ161及びデコーダ164は、上述のようにソフトウェアで構成されていてもよいし、あるいは専用ハードウェアで構成してもよい。

【0075】なお、上述の第1の実施の形態、すなわち図3のMIDIデータ編集処理1では、ステップ38において、前回のキーオンイベントデータのキーコードを格納した前回値レジスタb0と今回のキーオンイベントデータのキーコードを格納したキーコードレジスタbとの間の差分を取り、それをキーコード差分値レジスタdkeyに格納すると共に、前回のキーオンイベントデータのベロシティを格納した前回値レジスタc0と今回のキーオンイベントデータのベロシティを格納したベロシティレジスタcとの間の差分を取り、それをベロシティ差分値レジスタdvel1に格納している。その関係上、ステップ3Aからステップ3Gまでの処理によって変換された新しいキーコードの値b1及びベロシティの値c1は、常に前回のキーオンイベントにおけるキーコード及びベロシティとの間に密接な関係を有することにな

る。図４のＭＩＤＩデータ修復処理１でも、前回値レジスタｂ０とキーコードレジスタｂとに基づいて修復キーコードレジスタｂ２の値を求め、前回値レジスタｃ０とベロシティレジスタｃとに基づいて修復ベロシティレジスタｃ２の値を求めている。従って、図３のＭＩＤＩデータ編集処理後にＭＩＤＩデータのイベントデータ（キーコード又はベロシティの値）が修正されると、それ以降に修復処理された修復キーコード及び修復ベロシティの値が元のデータと全て異なってしまうということが起こる。これはデータを改竄した場合にそれ以降のデータを全て無効にすることができるという点では優れているが、途中のイベントデータにエラーが発生し、キーコード又はベロシティが変化した場合、それ以降のデータを有効に修復することができなくなるという問題を有する。

【００７６】そこで、図３のＭＩＤＩデータ編集処理１及び図４のＭＩＤＩデータ修復処理１を図１３のＭＩＤＩデータ編集処理４及び図１４のＭＩＤＩデータ修復処理４のように変更することによって、イベントデータが途中で書き換えられたりなどして変化した場合でも、それ以降のデータを有効に修復することができるようにした。図１３のＭＩＤＩデータ編集処理４において図３のＭＩＤＩデータ編集処理１と同じものには同一の符号が付してあるので、その説明は省略する。図１３のＭＩＤＩデータ編集処理４が図３のＭＩＤＩデータ編集処理１と異なる点は、図３のステップ３９の処理が省略され、ステップ３８の前回値レジスタｂ０及びｃ０がステップ１３８のように先頭値レジスタｂＨ及びｃＨに変更された点である。ステップ１３８では、先頭のキーオンイベントデータのキーコードを格納した先頭値レジスタｂＨと今回のキーオンイベントデータのキーコードを格納したキーコードレジスタｂとの間の差分を取り、それをキーコード差分値レジスタｄｋｅｙに格納すると共に、先頭のキーオンイベントデータのベロシティを格納した先頭値レジスタｃＨと今回のキーオンイベントデータのベロシティを格納したベロシティレジスタｃとの間の差分を取り、それをベロシティ差分値レジスタｄｖｅｌに格納する。そして、このキーコード差分値レジスタｄｋｅｙの格納値とベロシティ差分値レジスタｄｖｅｌの格納値に基づいて新たなキーコードｂ１及びベロシティｃ１を算出している。

【００７７】図１４のＭＩＤＩデータ修復処理４において図４のＭＩＤＩデータ修復処理１と同じものには同一の符号が付してあるので、その説明は省略する。図１４のＭＩＤＩデータ修復処理４が図４のＭＩＤＩデータ修復処理１と異なる点は、図４のステップ４８の処理が省略され、ステップ４５及びステップ４６の前回値レジスタｂ０及びｃ０がステップ１４５及びステップ１４６のように先頭値レジスタｂＨ及びｃＨに変更された点である。すなわち、ステップ１４５では先頭値レジスタｂＨとキーコードレジスタｂとの値に基づいて修復キーコードレジスタｂ２の値を求め、ステップ１４６では先頭値レジスタｃＨとベロシティレジスタｃとの値に基づいて修復ベロシティレジスタｃ２の値を求めている。図１３及び図１４のように変更することによって、先頭のイベントデータの値に基づいてＭＩＤＩデータは編集され、修復されるようになるので、途中のイベントデータが書き換えられたり、エラーなどの発生によって変化した場合でもそれ以降のデータを修復できなくなるというようなことは起こらず、エラーの発生したイベントデータ以降も完全に修復することができるようになる。なお、上述の説明では先頭のイベントデータの値に基づいてＭＩＤＩデータを編集する場合について説明したが、予め任意の値をヘッダに書き込んでおいて、その値に基づいてＭＩＤＩデータを編集したり、修復したりするようにしてもよい。また、その基準となるイベントデータの値を任意のイベント（例えば１小節における先頭のイベント）から取り込むようにしてもよい。

【００７８】ところで、図１４のＭＩＤＩデータ修復処理において、レジスタｂＨ，ｃＨにストアされるデータは、電子透しが埋め込まれた又は暗号化されたデータを修復するための「鍵情報」の一種に相当している。このデータ修復のための「鍵情報」は、上記実施例では先頭のキーオンイベントデータとして主要情報（電子透しまたは暗号化の対象となる情報）の中に含まれているものが用いられているが、これに限らず、データ修復のための「鍵情報」が、主要情報にも付属情報にも含まれないようにしてもよい。すなわち、図１４のＭＩＤＩデータ修復処理において、レジスタｂＨ，ｃＨにストアされるデータ修復用「鍵情報」を、主要情報及び付属情報のデータファイルとは別途に供給するようにする。勿論、その場合、レジスタｂＨ，ｃＨにストアされるデータ修復用「鍵情報」は、上記のような先頭のキーオンイベントデータのキーコード又はベロシティデータである必要はなく、適宜に設定したデータであってよい。そのためには、データ編集処理（つまり電子透し埋め込み又は暗号化すなわちエンコード処理）の際に、例えば図１３のステップ１３８で使用するレジスタｂＨ，ｃＨに格納するデータとして、所望のデータを任意に設定するようにすればよい。このレジスタｂＨ，ｃＨに格納するデータは前述の通り、電子透し埋め込み又は暗号化（つまりエンコード）のための所定のアルゴリズムにおいて変数の一種（エンコードしようとするデータの差分値を算出するための演算の変数）として、すなわち電子透し埋め込み又は暗号化演算のための補助情報として、使用されるものである。従って、所望の変数を設定入力し、これをレジスタｂＨ，ｃＨにストアすることで、該設定入力した変数を用いて電子透し埋め込み又は暗号化（つまりエンコード）処理が行われるようにすればよい。この変数（つまり、データ修復の際には「鍵情報」として利用さ

れるもの）の設定入力処理は、図１３のステップ１３８のあたりで行うようにしてもよいが、その前の段階、例えば、ステップ３１のあたりで、行うようにするとよい。勿論、この変数（つまり、データ修復の際には「鍵情報」として利用されるもの）の設定入力処理は、手動入力に限らず、データ通信その他適宜の手段を介して自動で行うようになっていてもよい。

【００７９】このようにすることにより、データ修復時においてレジスタｂＨ，ｃＨにストアすべきデータ修復用「鍵情報」を具備していない（若しくは知っていない）者が、不当にデータを修復しようとする場合、その修復アルゴリズムが判明していたとしても、肝心のデータ修復用「鍵情報」を得ることができないことにより、データ修復を行なうことができないことになり、暗号化又は電子透しによる機密情報埋め込みに際して、セキュリティ機能を向上させることができる。なお、この場合、レジスタｂＨ，ｃＨにストアされるデータ修復用「鍵情報」を主要情報及び付属情報のデータとは別途に供給する際の、該データ修復用「鍵情報」の供給の仕方は、どのようなやり方を用いてもよい。例えば、別途のデータ通信によって行なうようにしてもよいし、印刷物で通知してもよいし、電話等で通知してもよいし、様々なやり方であってよい。正当な利用者に対して秘密裡に通知又は供給されたデータ修復用「鍵情報」は、自動的に又は利用者による手動入力によってレジスタｂＨ，ｃＨにセットされるようになっていてよく、こうして、正当な利用者だけが図１４のようなデータ修復処理を正当に遂行し、データ修復を成功裡に行なうことができる。

【００８０】そのために、例えば、図１４に示すような「データ修復処理４」の手順を図１７に示す「データ修復処理４’」のように変更するとよい。図１７は、ステップ４１の前にステップ４０が挿入されている点だけが図１４とは異なっており、他は同一構成であってよい。このステップ４０において、上記のように適宜入力された所定の鍵情報を取り込み、それに基づき必要なデータを各レジスタｂＨ，ｃＨにロードする。なお、この実施例の変形例として、レジスタｂＨ，ｃＨにストアされるべきデータ修復用「鍵情報」の一部を、主要情報及び付属情報のデータファイルとは別途に供給するようにし、残りは主要情報及び付属情報のデータファイルの中に適宜含ませるようにしてもよい。その場合は、別途に供給された「鍵情報」の一部とデータファイルの中に含まれた残りの部分とを合成することで、レジスタｂＨ，ｃＨにストアされるべきデータ修復用「鍵情報」を生成する。その場合、鍵情報を合成するステップは少なくともステップ４５の手前に設定するものとし、合成したデータを各レジスタｂＨ，ｃＨにロードするものとする。

【００８１】ところで、図３のＭＩＤＩデータ編集処理１及び図１３のＭＩＤＩデータ編集処理４では、キーオンデータのキーコード及びベロシティの内容がＭＩＤＩデータの先頭から書き換えられてしまうので、ＭＩＤＩデータ修復処理を行わずにＭＩＤＩデータ再生処理を行った場合に、再生された楽音は、全くデタラメなものとなってしまい、その一部だけを試聴するというようなことができない。そこで、図３の変更例として、図１５のＭＩＤＩデータ編集処理５のように、ステップ３Ｂとステップ３Ｃとの間に編集条件を判定するステップ３Ｊを挿入し、特定の編集条件に合致した場合にＭＩＤＩデータにスクランブル処理を施すようにするとよい。すなわち、電子透かし埋め込み又は暗号化（すなわちエンコード）処理のためのアルゴリズムにおいて使用される変数の一種として、すなわち電子透し埋め込み又は暗号化演算のための補助情報として、「編集条件」設定情報が追加されることになる。なお、図３の変更例である図１３の場合も、図１５と同様に、ステップ３Ｂとステップ３Ｃとの間にステップ３Ｊを挿入するように変形し、電子透かし埋め込み又は暗号化（すなわちエンコード）処理のためのアルゴリズムにおいて使用される変数の一種として「編集条件」の変数を追加するようにしてよい。

【００８２】例えば、ステップ３Ｊの編集条件として、ＭＩＤＩデータの演奏開始から所定時間（例えば３０秒などの時間）を経過したか否かを判定するように設定する。これによって、ＭＩＤＩデータの最初の部分にはスクランブル処理が施されなくなるので、その部分のＭＩＤＩデータは通常のＭＩＤＩデータ再生処理によって再生されるようになる。一方、所定時間経過後にＭＩＤＩデータを通常のＭＩＤＩデータ再生処理で再生しようとしても、その部分にはスクランブル処理が施されているので、デタラメな楽音しか再生されなくなる。これによって、ＭＩＤＩデータの最初の部分を通常のＭＩＤＩデータ再生処理によって再生できる試聴可能なＭＩＤＩデータとすることができ、それ以降の残りの部分を試聴不可能なＭＩＤＩデータとすることができる。なお、この「編集条件」変数は、外部から任意に設定可能であり、時間の他にも、種々の条件を適宜組み合わせて適用してもよいことは言うまでもない。例えば、発音されない時間が所定時間以上になった場合とか、小節線データが所定数発生した場合とかの条件を単独で適用したり、これらの条件を種々組み合わせてもよいことはいうまでもない。

【００８３】また、例えば、ＭＩＤＩデータファイルの先頭から電子透し埋め込み又は暗号化のための編集操作を開始するまでのデータ数及びそこから実際に編集を行なったデータ数を「編集条件」変数として設定する（すなわちＭＩＤＩデータファイルのどの箇所に電子透し埋め込み又は暗号化のための編集操作を施したかを具体的に編集条件として設定する）ようにしてもよく、そのようにすれば、ＭＩＤＩデータファイルの任意の箇所に対して電子透し埋め込み又は暗号化のためのデータ編集操作を加えることができる。なお、所望の編集条件

の設定処理は、ステップ３Ｊで行ってもよいが、その前の段階、例えばステップ３１のあたり、で所望の編集条件を設定入力し、これを取り込むようにすればよい。また、設定する編集条件は一種類に限らず、複数種類であってもよい。

【００８４】なお、データ修復処理の際には、暗号化（つまりエンコード）の際に用いられた編集条件がどのようなものであるかを示す情報を「鍵情報」として使用することで、正しい編集条件「鍵情報」を持つものだけがデータ修復を行えるようにするとよい。例えば、これらの編集条件をユーザー毎にあるいはＭＩＤＩデータファイル毎に異ならせて設定してデータ編集処理（つまり電子透かし埋め込み又は暗号化のためのエンコード処理）を行った上で、それぞれの編集条件を示す情報を、個別のデータ修復用「鍵情報」として別途適宜に供給するようにし、正当なる利用者がデータ修復時に該データ修復用「鍵情報」を正当に取得して、その「鍵情報」が示す編集条件と修復すべきデータにおける実際の編集条件とが合致したときにデータ修復処理が遂行されるようにするとよい。このようにすることにより、データ修復のアルゴリズムは共通であっても、この「鍵情報」の多様化によって、電子透し埋め込み又は暗号化の形態を多様化させることができ、信頼性／セキュリティの向上を図ることができる。

【００８５】なお、この場合のデータ修復用「鍵情報」を別途に供給するやり方は、前述の場合と同様に、どのようなやり方を用いてもよい。また、データ修復処理に際しての編集条件の照合の仕方としては、例えば、データ修復しようとする利用者が所要の「鍵情報」を入力することによって編集条件を設定入力し、この設定入力された編集条件に該当するデータファイル箇所において図１５のステップ３Ｃ～３Ｆに示すデータ暗号化処理の逆算であるデータ修復（デコード）処理を行なうようにする。ここで、設定入力された編集条件が正しければ該データ修復処理では正しくデータ修復がなされることになり、正しくなければ該データ修復処理では正しいデータ修復が行なえないことになる。なお、この場合の「鍵情報」の入力の仕方も、前述と同様に手動入力に限らず自動的になされるようにしてもよい。図１８は、図４に示すような「データ修復処理１」を、編集条件を鍵情報として使用してデータ修復を行うのに適したものに変形した一例（「データ修復処理５」）を示す図である。図１８において、ステップ４１の手前にステップ４０’を挿入し、ステップ４１と４２の間にステップ４１０を挿入した点が図４とは相違しており、他は図４と同じである。ステップ４０’では、編集条件の「鍵情報」の入力を受け付け、これを取り込む。ステップ４１０では、取り込んだ「鍵情報」によって示される編集条件が成立したか否かを調べる。編集条件が成立していなければ、ステップ４２以降のデータ修復処理は行わずにステップ４Ｃに行き、そのＮＯからステップ４１に戻る。取り込んだ「鍵情報」によって示される編集条件が成立した場合は、ステップ４２～４Ｂで所要のデータ修復処理を行う。なお、他の「データ修復処理」（例えば図１４）に対しても、図１８と同様の変形を施すことができる。

【００８６】なお、上記のようにデータ修復用「鍵情報」は一種類とは限らず、複数種類あってよいものである。また、同じ種類の鍵情報でも異なる値のものが複数あってもよい。その場合、例えば鍵管理ファイルをデータファイルとは別途に用意しておき、この鍵管理ファイルから必要な「鍵情報」を正当な利用者だけが取得できるように管理するとよい。例えば、利用者は所定のアルゴリズムでデータファイルから鍵取得情報を抽出し、その鍵取得情報を入力情報として鍵管理ファイルから必要な鍵情報を取得できるようにする。この場合、鍵管理ファイルは入力された鍵取得情報に応じて適宜異なる鍵情報を供給するものとする。このようにして複数種類の必要な鍵情報をそれぞれ取得できるようにし、これらの鍵情報を使用して上記のようにデータ修復を行うようにする。ここで、更にデータファイル内の複数の箇所で複数回異なる鍵情報を取得しなければ適正なデータ修復が行なえないようにすれば、より厳密な管理とセキュリティ保持を行なうことができる。なお、鍵管理ファイルの部分は通信ネットワークのサーバー上に格納しておき、データ通信によってオーソライズされた利用者だけが鍵取得情報を入力して、これに応じてサーバーから必要な鍵情報をデータ通信によって随時取得することができるようにしてもよい。

【００８７】

【発明の効果】この発明によれば、音楽データ、映像データ又は波形データなど主要情報のデータフォーマットを変更することなく所望の付属情報を潜在的に付加してなるデータファイルを提供することができると共に結果的にこれらのデータに暗号化処理を施すこととなり、この暗号を解読しない限り、これらの主要情報や付属情報を再生して利用することができないという効果がある。また、主要情報を構成するデータ群の中の一部のデータを付属情報を構成するデータに基づいて変更するために、主要情報における変更すべき前記一部のデータに対して、所定のアルゴリズムに従う演算を、前記付属情報を構成するデータと外部から設定可能な補助情報とを用いて行なうようにしたことで、解読の際の鍵情報となりうる補助情報そのものは、データファイルのデータ内容とは無関係に任意に設定されるから、該データファイルの中には直接的にはそれと判るように含まれていないものとすることができるものであり、これにより、オーソライズされていない者が、不当に当該データファイルから主要情報の修復を行ないまたその付属情報を解読しようとしたとしても、肝心のデータ修復用の「鍵情報」をデータファイルからからは得ることができないことによ

り、データ修復若しくは解読を行なうことができないことになり、暗号化又は電子透し埋め込みによる機密的な情報記録に際して、セキュリティ面での信頼性を向上させることができる、という優れた効果を奏する。

【図面の簡単な説明】

【図１】　この発明に係る電子情報処理システムのＭＩＤＩデータ編集処理１及びＭＩＤＩデータ修復処理１に従ってどのようにデータが変換されるのか、その具体例を示す図である。

【図２】　この発明に係る電子情報処理システムとして動作する電子楽器の全体構成を示すブロック図である。

【図３】　図２の電子楽器が電子情報処理システムとして動作する場合のＭＩＤＩデータ編集処理１の一例を示すフローチャート図である。

【図４】　図２の電子楽器が電子情報処理システムとして動作する場合のＭＩＤＩデータ修復処理１の一例を示すフローチャート図である。

【図５】　縦軸にキーコードを割り当て、横軸にベロシティを割り当て、各値の中間を原点とするような直交座標系を構成した場合の付属情報とキーコード，ベロシティとの関係を示す概念図である。

【図６】　図５の付属情報とキーコード，ベロシティとの間の詳細な関係を示す図である。

【図７】　ＭＩＤＩデータ中に埋め込まれるスクランブルデコードデータの内容の一例を示す図である。

【図８】　電子楽器が電子情報処理システムとして動作する場合の別の一例となるＭＩＤＩデータ編集処理２を示すフローチャート図である。

【図９】　電子楽器が電子情報処理システムとして動作する場合の別の一例となるＭＩＤＩデータ修復処理２を示すフローチャート図である。

【図１０】　電子楽器が電子情報処理システムとして動作する場合の別の一例となるＭＩＤＩデータ編集処理３を示すフローチャート図である。

【図１１】　電子楽器が電子情報処理システムとして動作する場合の別の一例となるＭＩＤＩデータ修復処理３を示すフローチャート図である。

【図１２】　この発明に係る電子情報処理システムのＭＩＤＩデータ編集処理２及びＭＩＤＩデータ修復処理２に従ってどのようにデータが変換されるのか、その具体例を示す図である。

【図１３】　電子楽器が電子情報処理システムとして動作する場合の別の一例となるＭＩＤＩデータ編集処理４を示すフローチャート図である。

【図１４】　電子楽器が電子情報処理システムとして動作する場合の別の一例となるＭＩＤＩデータ修復処理４を示すフローチャート図である。

【図１５】　電子楽器が電子情報処理システムとして動作する場合の別の一例となるＭＩＤＩデータ編集処理５を示すフローチャート図である。

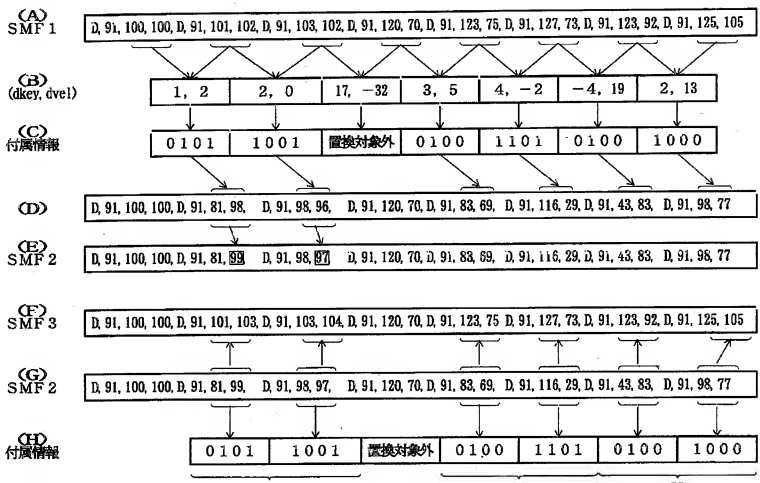
【図１６】　通信ネットワークを通じて実施する本発明の一実施の形態を例示するブロック図である。

【図１７】　図１４のＭＩＤＩデータ修復処理の変形例を示すフローチャート図である。

【図１８】　図１５のＭＩＤＩデータ編集処理５で処理された情報のデータ修復に使用できるＭＩＤＩデータ修復処理５の一例を示すフローチャート図である。

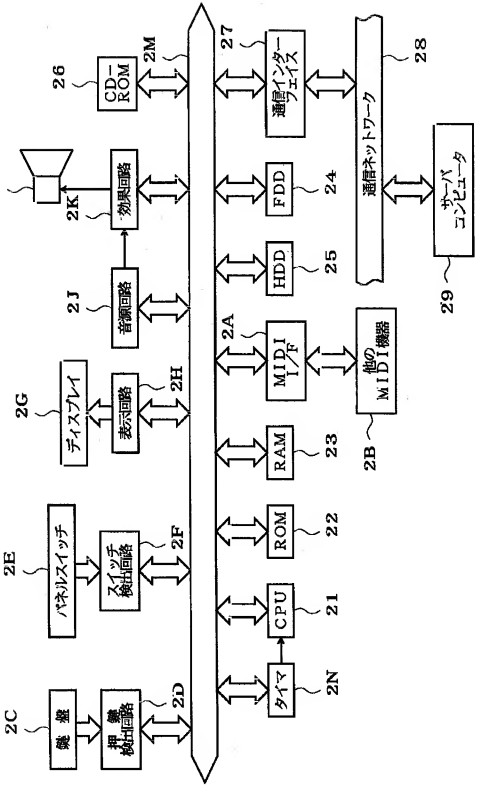
【符号の説明】

２１…ＣＰＵ、２２…ＲＯＭ、２３…ＲＡＭ、２４…フロッピーディスクドライブ、２５…ハードディスクドライブ、２６…ＣＤ－ＲＯＭドライブ、２７…通信インターフェイス、２８…通信ネットワーク、２９…サーバコンピュータ、２Ａ…ＭＩＤＩインターフェイス、２Ｂ…他のＭＩＤＩ機器、２Ｃ…鍵盤、２Ｄ…押鍵検出回路、２Ｅ…パネルスイッチ、２Ｆ…スイッチ検出回路、２Ｇ…ディスプレイ、２Ｈ…表示回路、２Ｊ…音源回路、２Ｋ…効果回路、２Ｌ…サウンドシステム、２Ｍ…データ及びアドレスバス、２Ｎ…タイマ、２Ｒ…リボンコントローラ

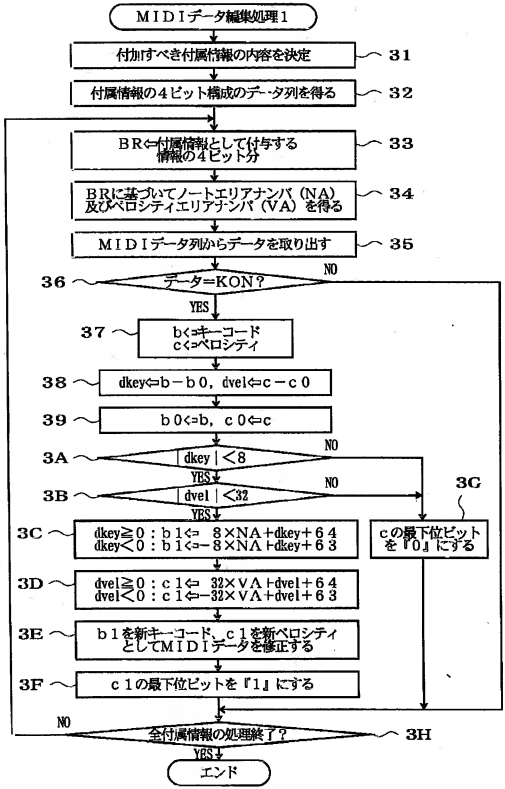

(A)
SMF 1
D, 91, 100, 100, D, 91, 101, 102, D, 91, 103, 102, D, 91, 120, 70, D, 91, 123, 75, D, 91, 127, 73, D, 91, 123, 92, D, 91, 125, 105
(B)
(dkey, dvel)
1, 2
2, 0
17, −32
3, 5
4, −2
−4, 19
2, 13
(C)
付属情報
0101
1001
置換対象外
0100
1101
0100
1000
(D)
D, 91, 100, 100, D, 91, 81, 98, D, 91, 98, 96, D, 91, 120, 70, D, 91, 83, 69, D, 91, 116, 29, D, 91, 43, 83, D, 91, 98, 77
(E)
SMF 2
D, 91, 100, 100, D, 91, 81, 98, D, 91, 98, 97, D, 91, 120, 70, D, 91, 83, 69, D, 91, 116, 29, D, 91, 43, 83, D, 91, 98, 77
(F)
SMF 3
D, 91, 100, 100, D, 91, 101, 103, D, 91, 103, 104, D, 91, 120, 70, D, 91, 123, 75 D, 91, 127, 73, D, 91, 123, 92, D, 91, 125, 105
(G)
SMF 2
D, 91, 100, 100, D, 91, 81, 99, D, 91, 98, 97, D, 91, 120, 70, D, 91, 83, 69, D, 91, 116, 29, D, 91, 43, 83, D, 91, 98, 77
(H)
付属情報
0101
1001
置換対象外
0100
1101
0100
1000
Y
M
H

【図1】

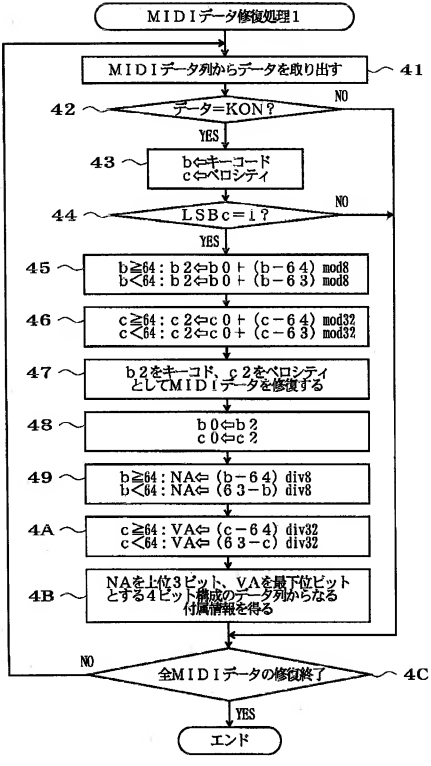
【図2】


2C
鍵盤
2D
押鍵検出回路
2E
パネルスイッチ
スイッチ検出回路
2F
2G
ディスプレイ
表示回路
2H
2J
音源回路
2K
効果回路
2L
26
CD-ROM
2M
27
通信インターフェイス
2A
MIDI I/F
2B
他のMIDI機器
HDD
25
FDD
24
通信ネットワーク
28
サーバコンピュータ
29
2N
タイマ
CPU
21
ROM
22
RAM
23

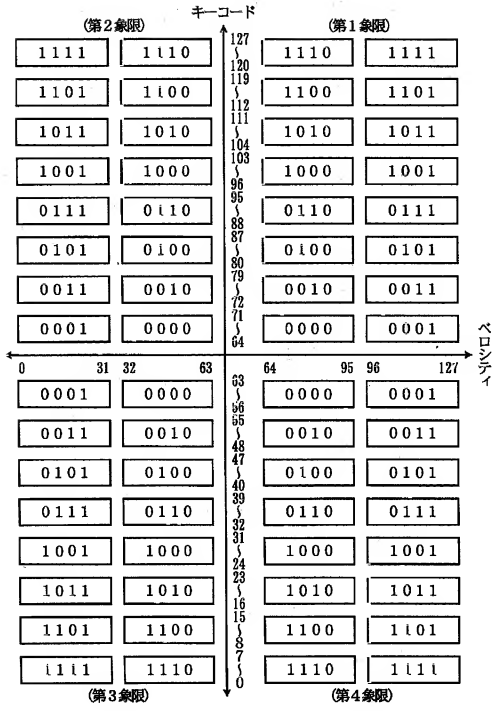
【図3】


MIDIデータ編集処理1
付加すべき付属情報の内容を決定
31
付属情報の4ビット構成のデータ列を得る
32
BR⇐付属情報として付与する
情報の4ビット分
33
BRに基づいてノートエリアナンバ（NA）
及びベロシティエリアナンバ（VA）を得る
34
MIDIデータ列からデータを取り出す
35
36
データ=KON？
NO
YES
37
b⇐キーコード
c⇐ベロシティ
38
dkey⇐b－b0，dvel⇐c－c0
39
b0⇐b，c0⇐c
3A
dkey｜＜8
NO
YES
3B
dvel｜＜32
NO
YES
3G
cの最下位ビット
を『0』にする
3C
dkey≧0：b1⇐ 8×NA＋dkey＋64
dkey＜0：b1⇐－8×NA＋dkey＋63
3D
dvel≧0：c1⇐ 32×VA＋dvel＋64
dvel＜0：c1⇐－32×VA＋dvel＋63
3E
b1を新キーコード、c1を新ベロシティ
としてMIDIデータを修正する
3F
c1の最下位ビットを『1』にする
NO
全付属情報の処理終了？
3H
YES
エンド

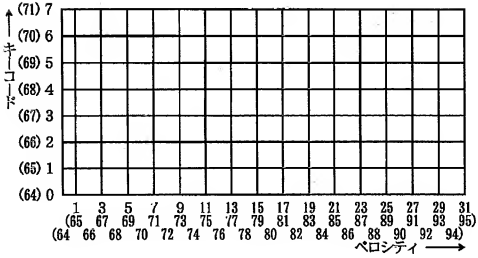
【図4】


MIDIデータ修復処理1
MIDIデータ列からデータを取り出す
41
42
データ=KON?
NO
YES
43
b⇐キーコード
c⇐ベロシティ
44
LSBc=1?
NO
YES
45
b≧64：b2⇐b0＋(b－64) mod8
b<64：b2⇐b0＋(b－63) mod8
46
c≧64：c2⇐c0＋(c－64) mod32
c<64：c2⇐c0＋(c－63) mod32
47
b2をキーコド、c2をベロシティ
としてMIDIデータを修復する
48
b0⇐b2
c0⇐c2
49
b≧64：NA⇐(b－64) div8
b<64：NA⇐(63－b) div8
4A
c≧64：VA⇐(c－64) div32
c<64：VA⇐(63－c) div32
4B
NAを上位3ビット、VAを最下位ビット
とする4ビット構成のデータ列からなる
付属情報を得る
NO
全MIDIデータの修復終了
4C
YES
エンド

【図5】


キーコード
(第2象限)
(第1象限)
(第3象限)
(第4象限)
ベロシティ
0
31
32
63
64
95
96
127
127〜120
119〜112
111〜104
103〜96
95〜88
87〜80
79〜72
71〜64
63〜56
55〜48
47〜40
39〜32
31〜24
23〜16
15〜8
7〜0
1111
1110
1101
1100
1011
1010
1001
1000
0111
0110
0101
0100
0011
0010
0001
0000

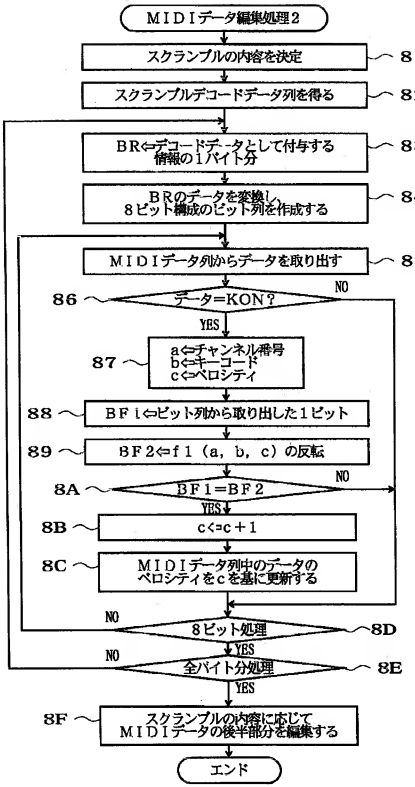
【図6】



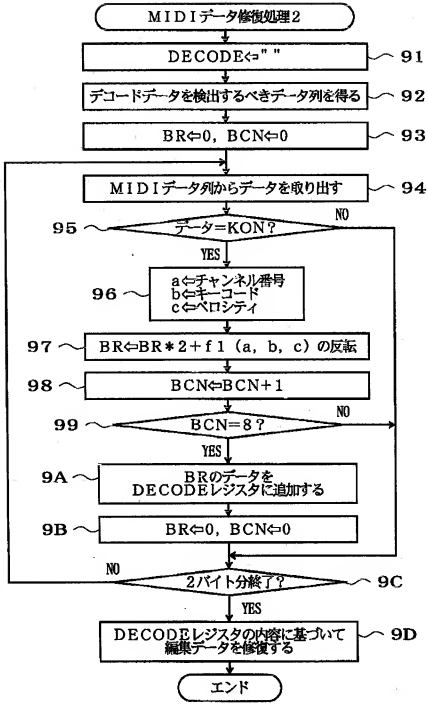
【図7】

4ビット	4ビット	8ビット
アルゴリズム	カウント数	Ｖａｌｕｅ
キーオンデータのノート番号を変更する	修正するイベント数	トランスポーズ値（修正ノート値）
キーオンデータのノートとベロシティを入れ替える	修正するイベント数	なし
インターバルデータを変更する	修正するイベント数	修正値
キーオンデータのチャンネルデータを変更する	修正するイベント数	修正チャンネル
その他		

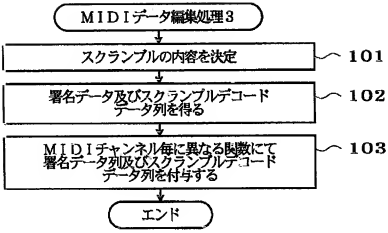
【図8】


MIDIデータ編集処理2
スクランブルの内容を決定
81
スクランブルデコードデータ列を得る
82
BR⇐デコードデータとして付与する情報の1バイト分
83
BRのデータを変換し、8ビット構成のビット列を作成する
84
MIDIデータ列からデータを取り出す
85
86
データ=KON?
NO
YES
87
a⇐チャンネル番号
b⇐キーコード
c⇐ベロシティ
88
BF1⇐ビット列から取り出した1ビット
89
BF2⇐f1（a，b，c）の反転
8A
BF1=BF2
NO
YES
8B
c⇐c＋1
8C
MIDIデータ列中のデータのベロシティをcを基に更新する
NO
8ビット処理
8D
YES
NO
全バイト分処理
8E
YES
8F
スクランブルの内容に応じてMIDIデータの後半部分を編集する
エンド

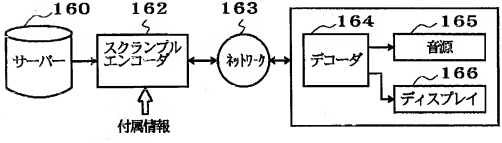
【図9】


MIDIデータ修復処理2
DECODE⇦""
91
デコードデータを検出するべきデータ列を得る
92
BR⇦0, BCN⇦0
93
MIDIデータ列からデータを取り出す
94
95
データ=KON?
NO
YES
96
a⇦チャンネル番号
b⇦キーコード
c⇦ベロシティ
97
BR⇦BR*2+f1 (a, b, c) の反転
98
BCN⇦BCN+1
99
BCN=8?
NO
YES
9A
BRのデータを
DECODEレジスタに追加する
9B
BR⇦0, BCN⇦0
NO
2バイト分終了?
9C
YES
DECODEレジスタの内容に基づいて
編集データを修復する
9D
エンド

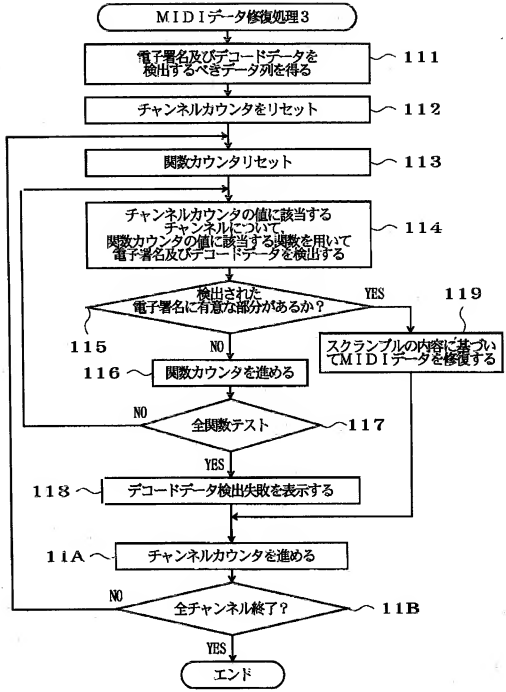
【図10】


MIDIデータ編集処理3
スクランブルの内容を決定
101
署名データ及びスクランブルデコード
データ列を得る
102
MIDIチャンネル毎に異なる関数にて
署名データ列及びスクランブルデコード
データ列を付与する
103
エンド

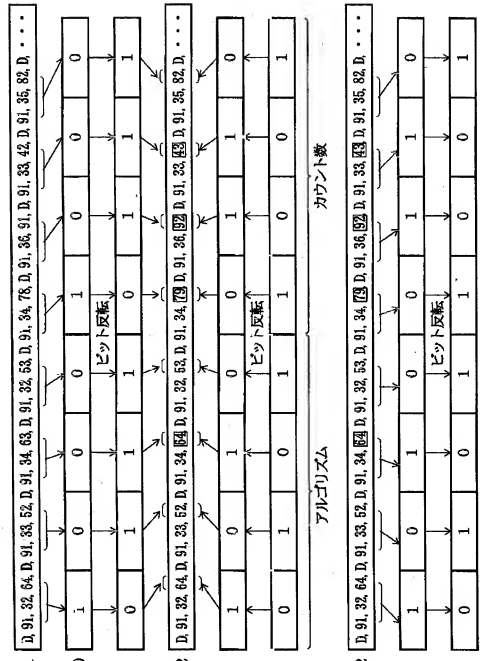
【図16】


160
162
163
164
165
166
サーバー
スクランブル
エンコーダ
ネットワーク
デコーダ
音源
ディスプレイ
付属情報

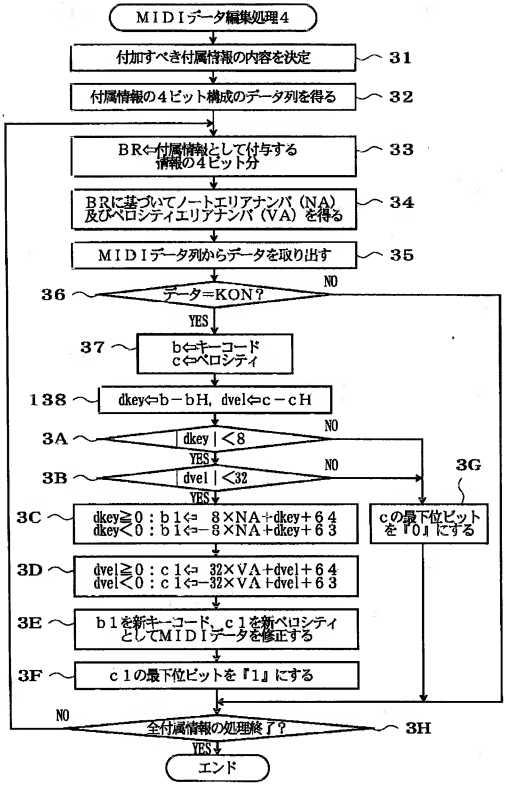
【図11】


MIDIデータ修復処理3
電子署名及びデコードデータを検出するべきデータ列を得る
111
チャンネルカウンタをリセット
112
関数カウンタリセット
113
チャンネルカウンタの値に該当するチャンネルについて、関数カウンタの値に該当する関数を用いて電子署名及びデコードデータを検出する
114
検出された電子署名に有意な部分があるか？
YES
119
NO
115
スクランブルの内容に基づいてMIDIデータを修復する
116
関数カウンタを進める
NO
全関数テスト
117
YES
118
デコードデータ検出失敗を表示する
11A
チャンネルカウンタを進める
NO
全チャンネル終了？
11B
YES
エンド

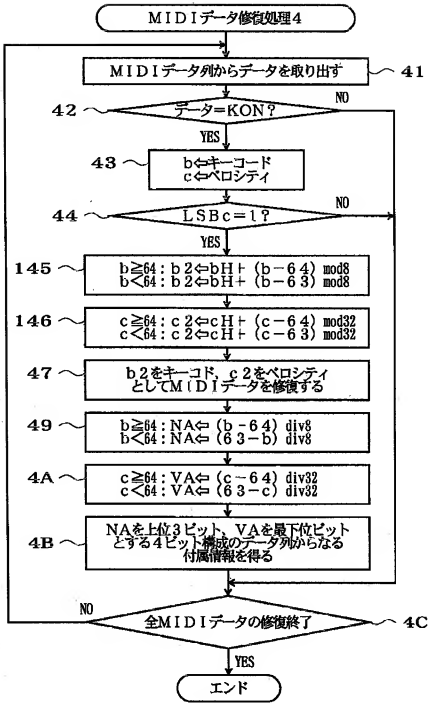
【図12】


(A) SMF 1
D, 91, 32, 64, D, 91, 33, 52, D, 91, 34, 63, D, 91, 32, 53, D, 91, 34, 76, D, 91, 36, 91, D, 91, 33, 42, D, 91, 35, 82, D, ・・・
(B) f1(a, b, c)
1 | 0 | 0 | 0 | 1 | 0 | 0 | 0
(C) BF2
0 | 1 | 1 | 1 | 0 | 1 | 1 | 1
ビット反転
(D) SMF 2
D, 91, 32, 64, D, 91, 33, 52, D, 91, 34, 63, D, 91, 32, 53, D, 91, 34, 76, D, 91, 36, 91, D, 91, 33, 42, D, 91, 35, 82, D, ・・・
(E) BF1
1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0
(F) BR
0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1
ビット反転
アルゴリズム
カウント数
(G) SMF 2
D, 91, 32, 64, D, 91, 33, 52, D, 91, 34, 63, D, 91, 32, 53, D, 91, 34, 76, D, 91, 36, 91, D, 91, 33, 42, D, 91, 35, 82, D, ・・・
(H) f1(a, b, c)
1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0
(I) BR
0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 1
ビット反転

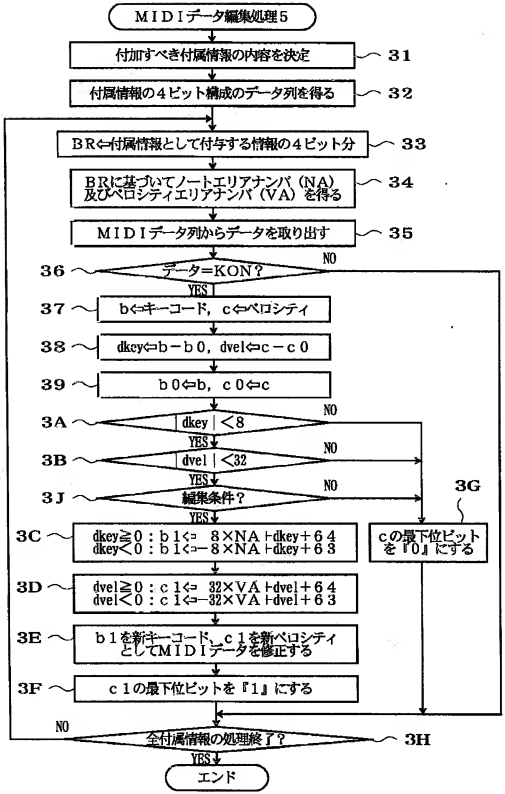
【図13】


MIDIデータ編集処理4
付加すべき付属情報の内容を決定
31
付属情報の4ビット構成のデータ列を得る
32
BR⇐付属情報として付与する情報の4ビット分
33
BRに基づいてノートエリアナンバ（NA）及びベロシティエリアナンバ（VA）を得る
34
MIDIデータ列からデータを取り出す
35
36
データ=KON?
NO
YES
37
b⇐キーコード
c⇐ベロシティ
138
dkey⇐b－bH, dvel⇐c－cH
3A
dkey|＜8
NO
YES
3B
dvel|＜32
NO
YES
3G
3C
dkey≧0：b1⇐ 8×NA＋dkey＋64
dkey＜0：b1⇐－8×NA＋dkey＋63
cの最下位ビットを『0』にする
3D
dvel≧0：c1⇐ 32×VA＋dvel＋64
dvel＜0：c1⇐－32×VA＋dvel＋63
3E
b1を新キーコード、c1を新ベロシティとしてMIDIデータを修正する
3F
c1の最下位ビットを『1』にする
NO
全付属情報の処理終了?
3H
YES
エンド

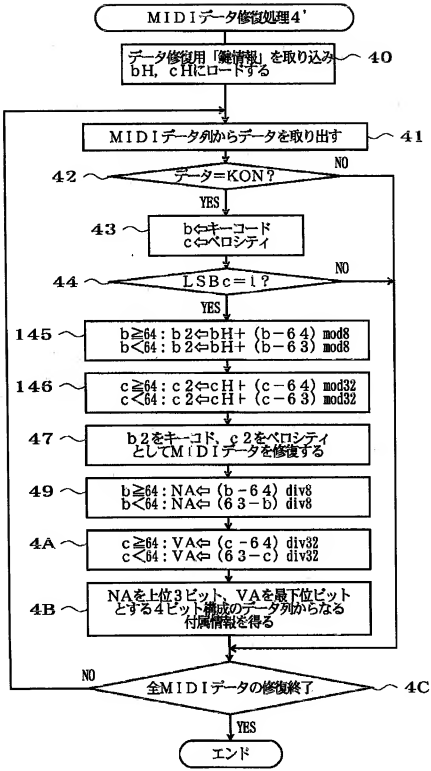
【図14】


MIDIデータ修復処理4
MIDIデータ列からデータを取り出す
41
42
データ=KON?
NO
YES
43
b⇐キーコード
c⇐ベロシティ
44
LSBc=1?
NO
YES
145
b≧64：b2⇐bH＋（b－64）mod8
b<64：b2⇐bH＋（b－63）mod8
146
c≧64：c2⇐cH＋（c－64）mod32
c<64：c2⇐cH＋（c－63）mod32
47
b2をキーコド，c2をベロシティ
としてMIDIデータを修復する
49
b≧64：NA⇐（b－64）div8
b<64：NA⇐（63－b）div8
4A
c≧64：VA⇐（c－64）div32
c<64：VA⇐（63－c）div32
4B
NAを上位3ビット，VAを最下位ビット
とする4ビット構成のデータ列からなる
付属情報を得る
NO
全MIDIデータの修復終了
4C
YES
エンド

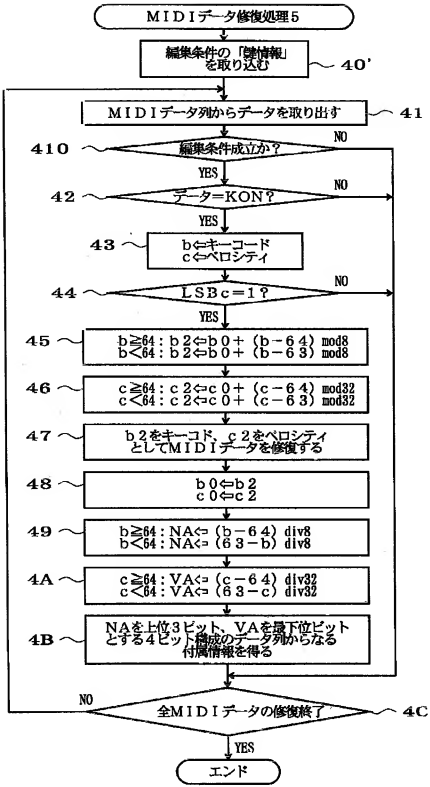
【図15】


MIDIデータ編集処理5
付加すべき付属情報の内容を決定
31
付属情報の4ビット構成のデータ列を得る
32
BR⇐付属情報として付与する情報の4ビット分
33
BRに基づいてノートエリアナンバ（NA）及びベロシティエリアナンバ（VA）を得る
34
MIDIデータ列からデータを取り出す
35
NO
36
データ=KON？
YES
37
b⇐キーコード，c⇐ベロシティ
38
dkey⇐b－b0，dvel⇐c－c0
39
b0⇐b，c0⇐c
NO
3A
dkey｜＜8
YES
NO
3B
dvel｜＜32
YES
NO
3J
編集条件？
3G
YES
dkey≧0：b1⇐ 8×NA＋dkey＋64
dkey＜0：b1⇐－8×NA＋dkey＋63
3C
cの最下位ビットを『0』にする
dvel≧0：c1⇐ 32×VA＋dvel＋64
dvel＜0：c1⇐－32×VA＋dvel＋63
3D
b1を新キーコード，c1を新ベロシティとしてMIDIデータを修正する
3E
c1の最下位ビットを『1』にする
3F
NO
全付属情報の処理終了？
3H
YES
エンド

【図17】


MIDIデータ修復処理4'
データ修復用「鍵情報」を取り込み bH，cHにロードする
40
MIDIデータ列からデータを取り出す
41
42
データ=KON?
NO
YES
43
b⇦キーコード
c⇦ベロシティ
44
LSBc＝1?
NO
YES
145
b≧64：b2⇦bH＋（b－64）mod8
b<64：b2⇦bH＋（b－63）mod8
146
c≧64：c2⇦cH＋（c－64）mod32
c<64：c2⇦cH＋（c－63）mod32
47
b2をキーコド、c2をベロシティとしてMIDIデータを修復する
49
b≧64：NA⇦（b－64）div8
b<64：NA⇦（63－b）div8
4A
c≧64：VA⇦（c－64）div32
c<64：VA⇦（63－c）div32
4B
NAを上位3ビット、VAを最下位ビットとする4ビット構成のデータ列からなる付属情報を得る
NO
全MIDIデータの修復終了
4C
YES
エンド

【図18】


MIDIデータ修復処理5
編集条件の「鍵情報」を取り込む
40'
MIDIデータ列からデータを取り出す
41
410
編集条件成立か？
NO
YES
42
データ=KON？
NO
YES
43
b⇐キーコード
c⇐ベロシティ
44
LSBc＝1？
NO
YES
45
b≧64：b2⇐b0＋（b－64）mod8
b<64：b2⇐b0＋（b－63）mod8
46
c≧64：c2⇐c0＋（c－64）mod32
c<64：c2⇐c0＋（c－63）mod32
47
b2をキーコード、c2をベロシティとしてMIDIデータを修復する
48
b0⇐b2
c0⇐c2
49
b≧64：NA⇐（b－64）div8
b<64：NA⇐（63－b）div8
4A
c≧64：VA⇐（c－64）div32
c<64：VA⇐（63－c）div32
4B
NAを上位3ビット、VAを最下位ビットとする4ビット構成のデータ列からなる付属情報を得る
NO
全MIDIデータの修復終了
4C
YES
エンド